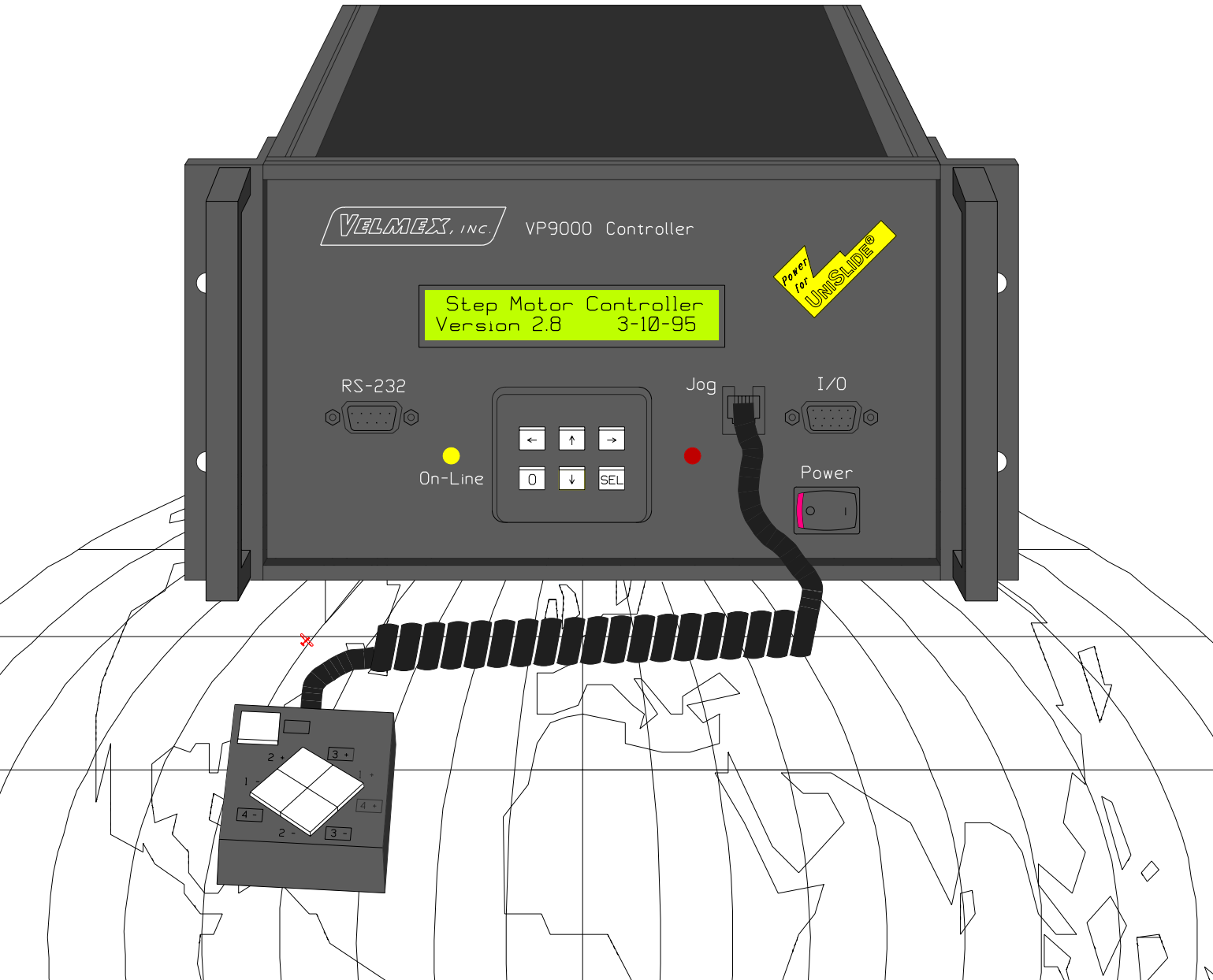
VELMEX, INC.
7550 State Routes 5 & 20
Bloomfield, NY 14469
Phone [716] 657-6151 Fax [716] 657-6153
Internet http://www.velmex.com

VP9000 SERIES

User's Guide

One, Two, Three, and Four-axis
Stepping Motor Controller/drivers



VP9000 Series Stepping Motor Controllers User's Guide

Velmex Incorporated, East Bloomfield, New York, USA

August 2 , 1996

Table of Contents

Introduction

The VP9000 is a programmable stepping motor controller, capable of running up to four motors, one-at-a-time. The Controller incorporates a powerful microprocessor and support circuitry, including 64K of nonvolatile RAM for storing programs and setup parameters.
Commands and data are entered either through the RS-232 interface or selected with the front panel keyboard. A 48 character alpha-numeric display, shows motor position(s), setup parameters and displays menus for keyboard selections.
Commands can be transmitted from a host computer, terminal, or programmable controller. Specialized commands provide simple and efficient entry of complex, yet compact, programs.

Features

- ✓ A complete microprocessor-based Controller with motor drives for one to four motors.
- ✓ Completely wired and tested for direct connection to UniSlide motors/ assemblies.
- ✓ High Power, efficient Bi-level Motor Drives.
- ✓ 400 steps per revolution (0.9º step angle) resolution
- ✓ Operates size 17 to size 34 permanent magnet stepping motors.
- ✓ Linear type motor and logic power supplies result in low Electromagnetic Interference (EMI).
- ✓ 90-130 VAC, 50/60 Hz operation. 190-260 VAC, 50/60 Hz for overseas operation is available.
- ✓ A Remote Jog Controller is included, which allows motors to be jogged manually one step or slewed up to 8000 steps/sec.
- ✓ Front Panel Readouts can be set to display motor position in "real" units for various Lead Screw pitches and Rotary Table gear ratios.
- ✓ A Digitizing function can be utilized with the front panel display, or with a host computer or terminal.
- ✓ A three wire serial port, conforming to EIA standard RS-232-C, allows a host to enter Commands (ASCII characters) and Data, Poll for status, and read Position information.

- ✓ A twelve-foot serial communication cable, and 25 to 9 pin adapter for PCs comes standard.

- ✓ The VP9000 can run in an interactive or stand-alone mode.

- ✓ Acceleration / Deceleration is programmable from 1,000 to 127,000 steps / sec^2 in 1,000 step / sec^2 increments.

- ✓ Speed is programmable from 1 step / 2 sec to 8000 steps / sec in 1 step / sec increments. Up to six speed changes on the fly are possible in Continuous Index mode.

- ✓ Incremental and Absolute Index distances are programmable from ±1 to ±16,777,215 steps.

- ✓ Programmable Home to Limit and Zero Position commands.

- ✓ The VP9000 stores 31 different programs (total of 1900 different moves), or an optional 62 programs (Version 2.8E, 3800 different moves).

- ✓ Jump-to a different program and Jump-to-and-return commands allow programs to be extensions and subroutines to other programs.

- ✓ Nine powerful Loop Commands provide from one to continuous repeat operations, performing simple functions like auto-reverse to raster scans and other complex X,Y matrix patterns.

- ✓ Programmable pauses from 100 milliseconds to 109 minutes in 100 millisecond intervals, and 0.1 millisecond to 6.5535 seconds in 0.1 millisecond intervals.

- ✓ Two User Outputs can be programmed to turn On and Off an external solid state relay, or interface to other logic level devices.

- ✓ Two User Inputs can be utilized in a program as WAITs for an external logic "low" level, external switch, or relay closure.

- ✓ Special Inputs provide interruption of WAIT or motion commands. An interrupt will decelerate a running motor to a stop, or terminate a WAIT command and transfer control to a user specific program.

- ✓ A user programmable counter can totalize over four billion counts and display total on the front panel.

- ✓ The VP9000 can be programmed to send a pulse or character at preset distances without stopping or slowing the motor.

- ✓ Backlash Compensation can be set to automatically finish every index approaching from the positive direction.

- ✓ RS-232 baud rate is settable to 9600, 4800, 1200, or 300.
- ✓ As many as 255 controllers can be "daisy-chained" together allowing the host to address each one from just one serial port.
- ✓ Limit Switches for CW and CCW directions are provided with plug-in connection to UniSlide limit switch assemblies. Limits can be used for "homing"; unused limit switch inputs can be software disabled.
- ✓ The VP9000 can be set to signal the host when a limit switch has been encountered.
- ✓ Automatic Power Down reduces power consumption by de-energizing the motors when at a standstill. Motor number one can be set to stay energized when at a standstill.
- ✓ Single Step mode is provided for debugging a program or as a controlled interrupt.
- ✓ The VP9000 can be polled for its status at any time; additionally a prompt ("^") is automatically sent to the host when a program has finished.
- ✓ Self testing with error messages displayed on front panel minimize troubleshooting.
- ✓ Compact Enclosure, rack mountable, with no fans or vent openings.
- ✓ Motor position can be read while motor is in motion.
- ✓ Optional Encoder Interface Module provides position verification (stall detection) using low cost 100 count encoders.
- ✓ Terminal/Editor, Diagnostic and Example Programs for MS-DOS PCs on a diskette are included.
- ✓ Front panel Main Menu can be locked out to prevent operator tampering of programs, or altering of setup parameters
- ✓ Two Year Limited Warranty.

Setup

*** * CAUTION * ***

* HAZARDOUS VOLTAGE, DO NOT REMOVE CONTROLLER'S PANELS
* DO NOT CONNECT OR DISCONNECT MOTOR(S) WHEN POWER IS "ON"
* HIGH TEMPERATURE, VP9000 SHOULD BE KEPT AT LEAST 6 INCHES FROM ANY OBJECTS
* AIR MUST CIRCULATE AROUND THE CONTROLLER
* NEVER USE IN AN EXPLOSIVE ENVIRONMENT
* IN INDUSTRIAL ENVIRONMENTS, THE VP9000 MUST BE PROTECTED TO PREVENT METAL PARTICLES FROM GETTING INTO SMALL OPENINGS

VP9000 controllers are factory wired with 10 foot cables for motor(s) and limit switch(es). A 12 foot RS-232 cable with 9 to 25 pin adapter for PCs, and a remote Jog controller with a 6 foot coil cord is included.

CAUTION: Only connect motors listed in the "Specifications" section (p.50).

1. Connect cables to motors and limit switches. Plug in remote Jog controller at the front panel connector labeled "Jog". If you are going to use a computer with the VP9000, connect the RS-232 cable between the front panel connector labeled "RS-232" and your computer's (usually COM1 or COM2 on a PC) Serial port. Refer to "Appendix D" (p.61) for cable and connector requirements for RS-232 connections to the VP9000.

CAUTION: Motor cables should **never** be bundled together with the Limit Switch, or any I/O cabling. **Never** put any of the VP9000's cables with power cables in a common electrical conduit or ducting. **Always** keep Limit Switch and I/O cables at least 3 inches from Motor and Power cables.

CAUTION: Maximum Allowable Motor Cable length is 50 feet (15.2 meters) with 20 AWG wire.

IMPORTANT: Limit Switch inputs require a **closed** circuit for the motor(s) to operate, ie, for motors to operate, limit switch cables must be connected to the UniSlide limit assemblies, or be disabled (see "Appendix E" (p.62) for electrical connections). Refer to "Appendix A" (p.51) for disabling limit switch inputs from the Setup Submenu.

2. The VP9000 is factory set to 9600 baud, 7 data, 2 stop bits, and even parity. To change the baud rate from the Setup Submenu refer to "Appendix A" (p.51).

CAUTION: Never connect or disconnect motors with the power on; this may result in severe damage to the motor or motor drive electronics.

3. Plug the VP9000 into a 120VAC outlet.

4. Turn on the VP9000 by pushing the right side of the rocker switch labeled "Power" located on the front panel.

The red light located to the left of switch will light, and the following sign-on message should flash on the front panel display after the word "UniSlide" scrolls back and forth across the display:

```
 Step Motor Controller
Version 2.8      3-10-95
```

If no motors have been selected (New Controller) the VP9000 will automatically enter the "Mtrsel" menu. Refer to "Appendix A" (p.51) for information on selecting the motor(s).

The display now shows the motor positions of every motor attached. These positions represent the total of accumulated steps from jog/slewing or indexing the motor(s) under program control. Refer to "Appendix A" (p.51) to change the units displayed to inches, millimeters, or degrees for a particular leadscrew or rotary table gear ratio.

The VP9000 is now ready to receive commands or be jogged/slewed from the remote Jog controller.

Jog/slew Mode

When the motor positions are displayed and the On-Line light is not lit, the VP9000 is in the Jog/slew mode. In the Jog/slew mode, pressing the "0" key on the front panel keyboard will zero all motor position registers. Using the remote Jog controller, each motor can be jogged a single step or slewed to full speed in either direction.

To Jog motor 1 positive (CW), press the 1+ button momentarily; the motor will move one step CW. To slew the motor maintain pressure on the 1+ button; the motor will accelerate to speed (last programmed speed for this motor; minimum of 31 steps/sec.) until the button is released. To go beyond the last speed setting (up to 8000) press the ↑ button on the front panel keyboard while slewing. **NOTE:** some motors may stall at the high speeds.

To Jog motor 1 negative (CCW), press the 1- button momentarily; the motor will move one step CCW. To slew the motor maintain pressure on the 1- button; the motor will accelerate to speed (last programmed speed for this motor; minimum of 31 steps/sec.) until the button is released. To go beyond the last speed setting (up to 8000) press the ↑ button on the front panel keyboard while slewing. **NOTE:** some motors may stall at the high speeds.

To Jog motor 2 positive (CW), press the 2+ button momentarily; the motor will move one step CW. To slew the motor maintain pressure on the 2+ button; the motor will accelerate to speed (last programmed speed for this motor; minimum of 31 steps/sec.) until the button is released. To go beyond the last speed setting (up to 8000) press the ↑ button on the front panel keyboard while slewing. **NOTE:** some motors may stall at the high speeds.

To Jog motor 2 negative (CCW), press the 2- button momentarily; the motor will move one step CCW. To slew the motor maintain pressure on the 2- button; the motor will accelerate to speed (last programmed speed for this motor; minimum of 31 steps/sec.) until the button is released. To go beyond the last speed setting (up to 8000) press the ↑ button on the front panel keyboard while slewing. **NOTE:** some motors may stall at the high speeds.

To Jog or slew motor 3, hold the ⎵ button down and actuate the motor 2 buttons.

To Jog or slew motor 4, hold the ⎵ button down and actuate the motor 1 buttons.

If a limit switch is encountered while jog/slewing, the words "ON LIMIT" will appear where motor position is displayed. For example, if motor 1 runs to the positive limit, the display will look like this:

```
1 ON LIMIT+
```

If the Unislide assembly(ies) does not have limit switches, refer to "Appendix A" (p.51) for disabling limit switch inputs from the Setup Submenu.

Digitizing With a Host

The VP9000 stores its absolute position (relative to when registers were zeroed) in memory. The absolute registers reflect the accumulated distance from operating the motors in the Jog/slew mode and/or under program control. These registers can hold from -8,388,608 to +8,388,607 steps.

With a host terminal or computer connected via the RS-232 interface, the VP9000 can be used as a digitizer. In the Jog/slew mode the VP9000 will send motor position when it receives a "**D**" from the host.
Here is an example of what the host would receive when Motor 1 is at absolute 201, Motor 2 is at absolute -1294010, Motor 3 is at absolute 0, and Motor 4 is at 80000:

<lf>**X+0000201**<cr>
<lf>**Y-1294010**<cr>
<lf>**Z+0000000**<cr>
<lf>**T+0080000**<cr>

<lf> is a linefeed, <cr> is a carriage return.
The host can null ("zero") the registers by sending a "**N**" to the Controller.

Programming from the Front Panel

When the VP9000 is off-line (On-Line light is not lit) it can be programmed from the front panel. Front panel commands are entered by selecting the desired command and parameters from the front panel display using the "SEL" key. The →, ←, ↑, ↓, keyboard keys are used to move between menus and desired values.

Main Menu

To enter commands from the front panel the Main Menu must first be selected. To access the Main Menu press either the "SEL", ←, →, or ↓ key. The display will now look like this:

```
Pgm#  Accel Speed  Index
Pause I/O Loop Jmp Setup
```

Note that the blinking cursor is over the "P" of the "Pgm#" selection. If the "SEL" key is now pressed the Program Submenu will appear on the display. Use the arrow keys to move around the main menu, and use the "SEL" key to select a submenu.

To lockout operator access to the Main Menu, refer to "Appendix G" (p.64).

Program Submenu

The Program Submenu is selected by pressing the "SEL" key when the cursor is over the "P" of "Pgm#". The Program Submenu will look like this:

```
Pgm= PM   Del is - Pgm00
123456789-   Ent Run New
```

NOTE:

The message "Del is -" on the first line is a reminder that when a negative program (PM-x) number is selected that this is to delete the selected program number (x).

Acceleration Submenu

The Acceleration Submenu is selected by pressing the "SEL" key when the cursor is over the "A" of "Accel". The Acceleration Submenu will look like this:

```
Accel=A           Pgm00
123456789 M  Ent     New
```

Speed Submenu

The Speed Submenu is selected by pressing the "SEL" key when the cursor is over the "S" of "Speed". The Speed Submenu will look like this:

```
Speed=S            Pgm00
123456789 MA Ent     New
```

Index Submenu

The Index Submenu is selected by pressing the "SEL" key when the cursor is over the "I" of "Index". The Index Submenu will look like this:

```
Index=I            Pgm00
123456789-MA Ent     New
```

Pause Submenu

The Pause Submenu is selected by pressing the "SEL" key when the cursor is over the "P" of "Pause". The Pause Submenu will look like this:

```
Pause=P            Pgm00
123456789- A Ent     New
```

I/O Submenu

The I/O Submenu is selected by pressing the "SEL" key when the cursor is over the "I" of "I/O". The I/O Submenu will look like this:

```
I/O = U            Pgm00
123456789    Ent     New
```

Loop Submenu

The Loop Submenu is selected by pressing the "SEL" key when the cursor is over the "L" of "Loop". The Loop Submenu will look like this:

```
Loop= L            Pgm00
123456789-MA Ent     New
```

Jump Submenu

The Jump Submenu is selected by pressing the "SEL" key when the cursor is over the "J" of "Jmp". The Jump Submenu will look like this:

```
Jump= J            Pgm00
123456789 M  Ent     New
```

Setup Submenu

The Setup Submenu is selected by pressing the "SEL" key when the cursor is over the "S" of "Setup". The Setup Submenu will look like this:

```
Mtrsel RS-232 Bcomp Encd
Trq1 Lmts Scale Defaults
```

Refer to "Appendix A" (p.51) for making selections from the Setup Submenu.

Entering Values for Submenus

On the first line, all submenus except Setup will display the command name followed by an equal sign, and the command letter(s). This command letter(s) is the same letter(s) that a host would send to the VP9000 over the RS-232 interface. To the far right on the first line is the current program number. The program number can be from 00 to 31. The actual number is determined by the value entered in the Program Submenu or by a host over the RS-232 Interface.

The second line of a submenu has the selectable numerals 1 through 9. Zero is selected by the "0" key on the keyboard. Some submenus have selections such as, "-", "M", "A", and "Run" ("Run" is on the Program Submenu only) on the second line also. The "Ent" selection is on this line, it is used for finalizing the values selected and entering this command along with the selected value into memory. At the far right on the second line is the amount of memory remaining for the current program. When a program has 256 bytes available (no commands entered) then the word "New" will be displayed.

Example entry:

To program an Index on Motor 1 for negative 400 steps:

1. Select the Index Submenu from the Main Menu.
2. Press "SEL" (selects 1 for motor 1).
3. Press the → until the blinking cursor is over "M".
4. Press the "SEL" key.
5. Press the ← until the blinking cursor is over "-".
6. Press the "SEL" key.
7. Press the ← until the blinking cursor is over the number 4.
8. Press the "SEL" key.
9. Press and release the "0" key twice.

The first Line should show: **`Index=I1M-400  Pgm00`**

10. Press "SEL" (cursor is on "Ent")

NOTE:

Pressing the ↑ or ↓ key will delete any partial selections prior to selecting "Ent". Pressing the ↑ or ↓ key again or when no selections were made will cause an exit from the current Submenu back to the Main Menu.

To run this program (Run can also be initiated by a remote input, refer to "Appendix C" (p.59) for connections to I/O):

1. Select the Program Submenu from the Main Menu.
2. Press the ← key until the cursor is over "Run".
3. Press the "SEL" key.

Programming Shortcut:

The motor designation in Acceleration, Speed, and Index commands is optional if the desired motor has already been set as the current motor. The current motor is motor 1 when the Controller is first turned on. The last motor jog/slewed will be the current motor number. The current motor will be the number used in the last Acceleration, Speed, or Index command. Users that have only a one motor VP9000 (Model VP9001) do not have to use the motor designation in a command. For example, these commands would always be motor 1 commands of a one motor VP9000:

`A2,S4000,I400,`

For running a particular motor of a multi-motor VP9000, only the first Command needs the motor number. For example, all of these commands would be for motor 2:

`I2M200,I-200,S2000,IA0,`

Command Summary (RS-232 and Front Panel)

These commands are applicable to programming from the front panel and from a host over the RS-232 interface. For front panel programming, the first letter corresponds to the letter after the equal sign on the first line of the display (except for the Program Submenu which is "PM"). When programming over the RS-232 interface, these commands are ended with a carriage return (Enter key), comma, or period.

I*m***M***x*	Set steps to incremental **I**ndex motor CW (positive), *m*= motor# (1,2,3,4), *x*=1 to 16,777,215
I*m***M-***x*	Set steps to incremental **I**ndex motor CCW (negative), *m*= motor# (1,2,3,4), *x*=1 to 16,777,215
IA*m***M**x	Set **A**bsolute **I**ndex distance, m=motor# (1,2,3,4), x= ±1 to ±16,777,215 steps
IA*m***M0**	**I**ndex motor to **A**bsolute zero position, *m*=motor# (1,2,3,4)
IA*m***M-0**	Zero motor position for motor# *m*, *m*= 1,2,3,4
I*m***M0**	**I**ndex motor until positive limit is encountered, *m*=motor# (1,2,3,4)
I*m***M-0**	**I**ndex motor until negative limit is encountered, *m*=motor# (1,2,3,4)
S*m***M***x*	Set **S**peed of motor (70% power applied to motor), *m*= motor# (1,2,3,4), *x*=0 to 8000 steps / sec. 0 being 1 step / 2 sec.
SA*m***M***x*	Set **S**peed of motor and **A**scend to 100% power, *m*= motor# (1,2,3,4), *x*=0 to 8000 steps / sec. 0 being 1 step / 2 sec.
A*m***M***x*	**A**cceleration / deceleration, *m*= motor# (1,2,3,4), *x*=1 to 127; 1 is 1000 steps / sec^2, 2 is 2000 steps / sec^2, 127 is 127000 steps / sec^2, etc.
L0	**L**oop continually from the beginning or Loop-to-marker of the current program
LM0	Sets the **L**oop-to-marker at the current location in the program
LM-0	Resets the **L**oop-to-marker to the beginning of the current program
L*x*	**L**oop from beginning or Loop-to-marker *x*-1 times (*x*=2 to 65,535), when the loop reaches its last count the non-loop command directly preceding will be ignored
L-*x*	**L**oop from beginning or Loop-to-marker *x*-1 times, alternating direction of motor 1, when the loop reaches its last count the non-loop command directly preceding will be ignored
LA*x*	**L**oop **A**lways from beginning or Loop-to-marker *x*-1 times (*x*=2 to 65,535)
LA-*x*	**L**oop **A**lways from beginning or Loop-to-marker *x*-1 times, alternating direction of motor 1
LM-2	**L**oop once from beginning or Loop-to-marker reversing index direction of motor 2
LM-3	**L**oop once from beginning or Loop-to-marker reversing index direction of motor 1 and motor 2
P*x*	**P**ause *x* tenths of a second (*x*=0 to 65,535, 10 µsec pause when *x*=0)
PA*x*	**P**ause *x* tenths of a second **A**ltering output 1 high for duration of the pause (*x*=0 to 65,535, 10 µsec pause when *x*=0)
P-*x*	**P**ause *x* tenths of a millisecond (*x*=1 to 65,535)
PA-*x*	**P**ause *x* tenths of a millisecond **A**ltering output 1 high for duration of the pause (*x*=1 to 65,535)
PM*x*	Select **P**rogra**m** number *x*, *x*= 0 to 30
PM-*x*	Select and clear all commands from **P**rogra**m** number *x*, *x*= 0 to 30

J*x* **J**ump to the beginning of program number *x*, *x*= 0 to 30

JM*x* **J**ump to the beginning of program number *x* and come back for **M**ore after program *x* ends, *x*= 0 to 30

U0 Wait for a "high" on user input 1

U1 Wait for a high on user input 1, holding user output 1 high while waiting

U2 Enable Jog mode while waiting for an input

U3 Disable Jog mode while waiting for an input

U4 User output 1 "low" (reset state)

U5 User output 1 high

U6 Send "W" to host and wait for a "G" to continue

U7 Start of Continuous Index with pulse on output 2

U8 Start of Continuous Index sending "@" to the host

U9 End of Continuous Index

U10 Wait for a low on input 2

U11 Wait for a low on input 2, holding user output 2 high while waiting

U12 Wait for a keyboard key to jump to a program or continue:

← key to jump to program #21
→ key to jump to program #22
↓ key to jump to program #23
↑ key to jump to program #24
0 or SEL key to proceed in current program

U13 Wait for a button on Remote Control to jump to a program or continue:

1- key to jump to program #25
1+ key to jump to program #26
2- key to jump to program #27
2+ key to jump to program #28
▭ key to proceed in current program

U14 **U**ser output 2 low (reset state)

U15 **U**ser output 2 high

U16 Wait for ← key to be pressed on front panel keyboard

U17 Wait for → key to be pressed on front panel keyboard

U22 Wait for a keyboard key to jump to a program and come back, or continue:

← key to jump to program #21 and come back
→ key to jump to program #22 and come back
↓ key to jump to program #23 and come back
↑ key to jump to program #24 and come back
0 or SEL key to proceed in current program

U23 Wait for a button on Remote Control to jump to a program and come back, or continue:

1- key to jump to program #25 and come back
1+ key to jump to program #26 and come back
2- key to jump to program #27 and come back
2+ key to jump to program #28 and come back
▭ key to proceed in current program

U30 Wait for a high to low transition on user input 1
U31 Wait for a high to low transition on user input 1, holding user output 1 high while waiting
U32 Wait for 1- button to be pressed on Remote Control
U33 Wait for 1+ button to be pressed on Remote Control

U40 Wait for a low to high transition on input 2
U41 Wait for a low to high transition on input 2, holding user output 2 high while waiting

U50 Wait for a high and low on user input 1 with debouncing for a mechanical push-button switch
U51 Wait for a high and low on user input 1 with debouncing for a mechanical push-button switch, holding user output 1 high while waiting
U60 Wait for a low and high on user input 2 with debouncing for a mechanical push-button switch
U61 Wait for a low and high on user input 2 with debouncing for a mechanical push-button switch, holding user output 2 high while waiting
U70 Zero counts in user programmable counter
U71 Display user programmable counts "on"
U72 Display user programmable counts "off"
U73 Increment user programmable counter by one
U90 Wait for a high and low on Run connection I/O,4 with debouncing for a mechanical push-button switch

On-Line Mode/ Programming from a Host

Remote programming of the VP9000 is accomplished by sending commands (ASCII characters) to the VP9000 through the RS-232 interface. The simplest method to send commands is with a terminal or computer operating with a terminal program, such as VPTERM included on the Utility Disk, or the Terminal program in Microsoft Windows. Integrated into VPTERM is an editor for listing and modifying VP9000 programs. Make sure your terminal or terminal program is set at 7 data, 2 stop bits and even parity, no flow control, and 9600 baud.

Another method to send commands is with commercially available languages such as BASIC, C, PASCAL, FORTRAN, or ASSEMBLY.

To put the VP9000 in the On-Line mode, the host must send an "**E**", "**F**", or "**G**". When the Controller receives an "**E**","**F**", or "**G**" (while in the Jog/slew mode only) the On-line light will light.

The "**E**" puts the VP9000 on-line with echo "on" (echoes all characters received back to the host). The "**F**" puts the VP9000 on-line with echo "off". The "**G**" puts the VP9000 on-line with echo "off" and "tells" the VP9000 to add a carriage return when it finishes a transmission.
If you are using a terminal to communicate to the VP9000 use the "**E**" so that the characters the Controller echoes will be displayed on the terminal screen.

Command Summary (RS-232 Exclusive)

All of the previous commands listed are applicable to programming from the VP9000's front panel and over the RS-232 interface. Additionally, the following commands are only applicable to programming from a host over the RS-232 interface, they are ended with a carriage return (Enter key), comma, or period.

B*x* **B**acklash compensation, compensation on when *x*=1, off when *x*=0
O*x* Indicate limit switch **O**ver-travel to host, off when *x*=0, VP9000 sends "O" when *x*=1 and a limit switch is encountered

PM Request the number of the current **P**rogra**m**

The following commands do not need an ending carriage return, comma, or period:

Q **Q**uit On-Line mode (return to Jog/Slew mode)
R **R**un currently selected program
N **N**ull (zero) motors 1,2,3,4 absolute position registers
K **K**ill operation in progress and reset user outputs
V **V**erify Controller's status, VP9000 sends "B" to host if busy, "R" if ready, or "J" if in the Jog/slew mode
C **C**lear all commands from currently selected program
D **D**ecelerate to a stop (interrupts current index in progress)
E **E**nable On-Line mode with echo "on"
F Enable On-Line mode with echo "o**FF**"
G Enable On-Line mode with echo off **G**rouping a carriage return with the "^", "<", ">", "=", "**:**", "**W**", "**O**" responses; Also used as **G**o after waiting or holding
H Put Controller on **H**old (single step mode)
X Send position of motor 1 to host
Y Send position of motor 2 to host
Z Send position of motor 3 to host
T Send position of motor 4 to host
M Request **M**emory available for currently selected program
Request the **number** of the currently selected motor
% Requests encoder/motor status from last index, VP9000 sends:
- **=** when motor counts and encoder counts are **equal**
- < when encoder counts are **less** than motor counts (motor stall)
- > when encoder counts are **greater** than motor counts (motor overshoot)

***** Request the position when the last motor was interrupted by a "**D**" command or user input 4

The following are for VP9000s that are daisy-chained together:

{*x***}** Send commands to the next VP9000 in the "chain", *x* are any of the above commands
& Enable multiple VP9000s that are daisy-chained

Example Programs

These examples are included on the Utility Disk for RS-232 programming

Most of the following examples are applicable to front panel and RS-232 programming. Under each example there will be "RS232" indicating that this example is a RS-232 example, and "Keyb" if that example also applies to front panel programming.

The following examples are commands (ASCII characters) that can be sent to the VP9000 with a terminal or computer operating with a terminal program (such as VPTERM included on the Utility Disk). Make sure your terminal or terminal program is set at 7 data, 2 stop bits and even parity, and the same baud rate (baud rate is factory set to 9600) as the VP9000 (see "Appendix A" (p.51) to change baud rate on the VP9000). The other method to send commands is with commercially available languages such as BASIC, C, PASCAL, FORTRAN, or ASSEMBLY.

The "<cr>" is a carriage return character (<Enter> key on most keyboards). Command characters are in **LARGE BOLD.**

Example #1 RS232	Motors run	RAM usage (bytes)	Function
On-Line	-	-	Enable On-Line mode with echo on (On-Line light lit)

E

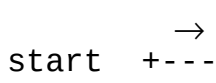
Example #2 RS232,Keyb	Motors run	RAM usage (bytes)	Function
Index	1	4	Incremental Index Motor 1 400 steps (1 rev) CW

I1M400,R

or

I1M400<cr>
R

Graphic Representation:



Front Panel Programmers: When programming from the front panel, the "R" represents "Run" on the Program Submenu, Run can also be initiated by a remote input, refer to "Appendix C" (p.59) for connections to I/O. The "," or <cr> represents the "Ent" selection from a submenu.

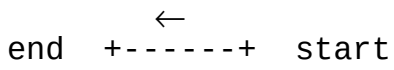
Example #3 RS232	Motors run	RAM usage (bytes)	Function
Clear	-	-	Clear all commands from the current program

C

Front Panel Programmers: To clear a program from the front panel keyboard, first select the Program Submenu from the Main Menu, and from the Program Submenu select "-" (negative) followed by the program number to clear.

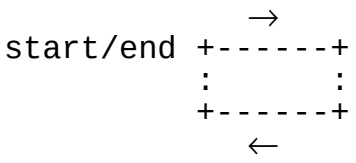
Example #4 RS232,Keyb	Motors run	RAM usage (bytes)	Function
Index	1	4	Incremental Index Motor 1 600 steps CCW

I1M-600,R



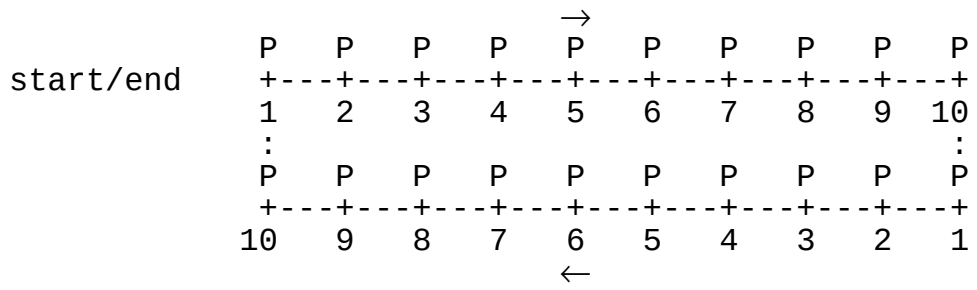
Example #5 RS232,Keyb	Motors run	RAM usage (bytes)	Function
Auto-Reverse	1	8	Auto-Reverse (motor 2)

I2M600,I2M-600,R



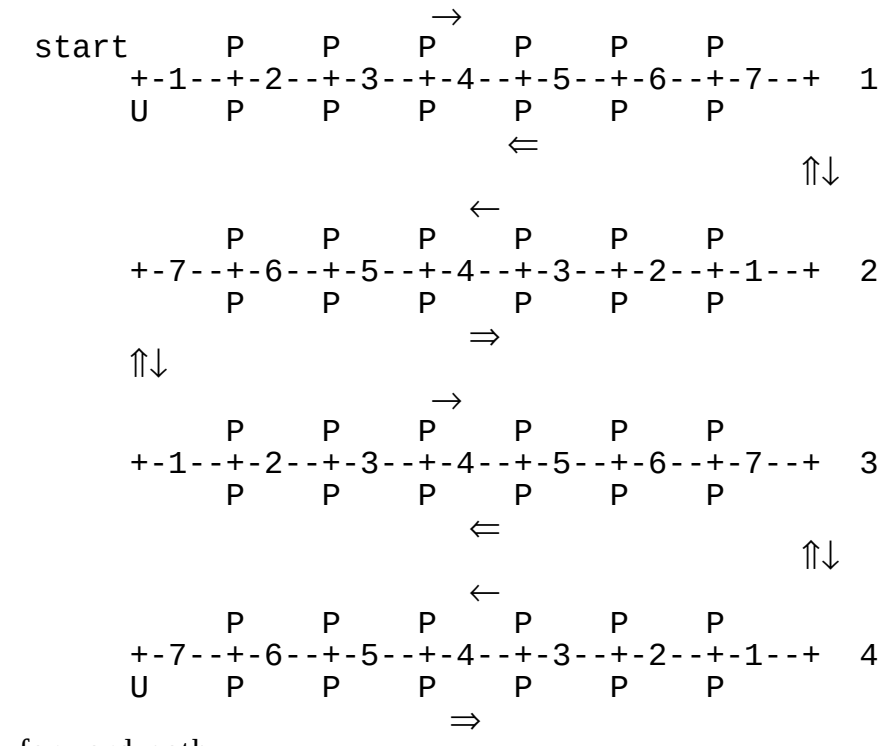
Example #6 RS232,Keyb	Motors run	RAM usage (bytes)	Function
Repeating Index	1	13	Repeating Index in two directions, pausing 1 sec. between Indexes

P10,I1M400,L10,L-2,R



Example #7 RS232,Keyb	Motors run	RAM usage (bytes)	Function
Raster Scan	2	23	Raster scan with 1 sec. pauses and waiting for input at beginning and the end, then run backwards through raster scan

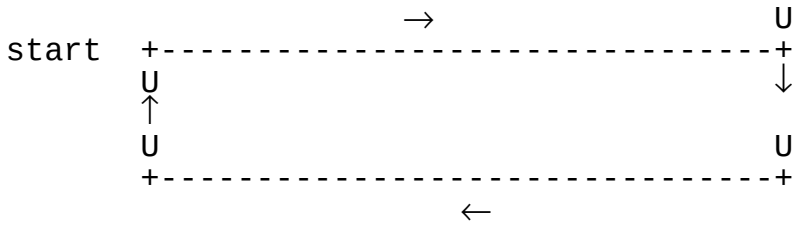
I1M200,P10,L7,I2M400,L-4,U0,LM-2,U0,L0,R



Key: → =forward path
⇒ =backwards path

Example #8 RS232,Keyb	Motors run	RAM usage (bytes)	Function
Rectangle	2	14	Rectangle, with Output and Wait at each corner

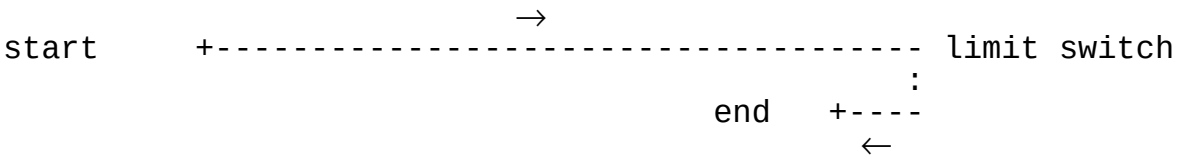
I1M2000,U1,I2M1000,U1,LM-3,L0,R



Example #9 RS232,Keyb	Motors run	RAM usage (bytes)	Function
Home to Limit	1	15	Home Motor 1 to Positive Limit Switch and move 200 steps from Limit Switch and zero position

S1M600,I1M0,I1M-200,IA1M-0,R

CAUTION: Motor speed should not be set above 1000 steps/sec. when homing to a limit switch.

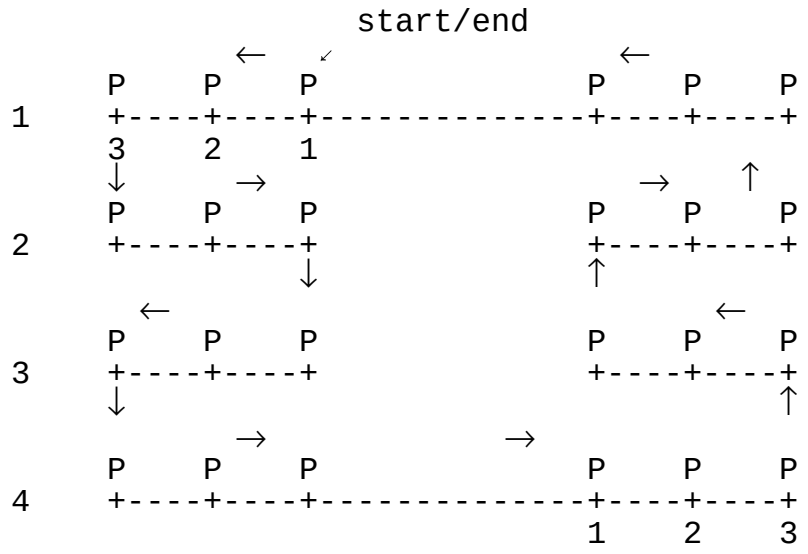


This would home Motor 1 to the negative Limit Switch and zero position 200 steps from the Limit Switch:

S1M600,I1M-0,I1M200,IA1M-0,R

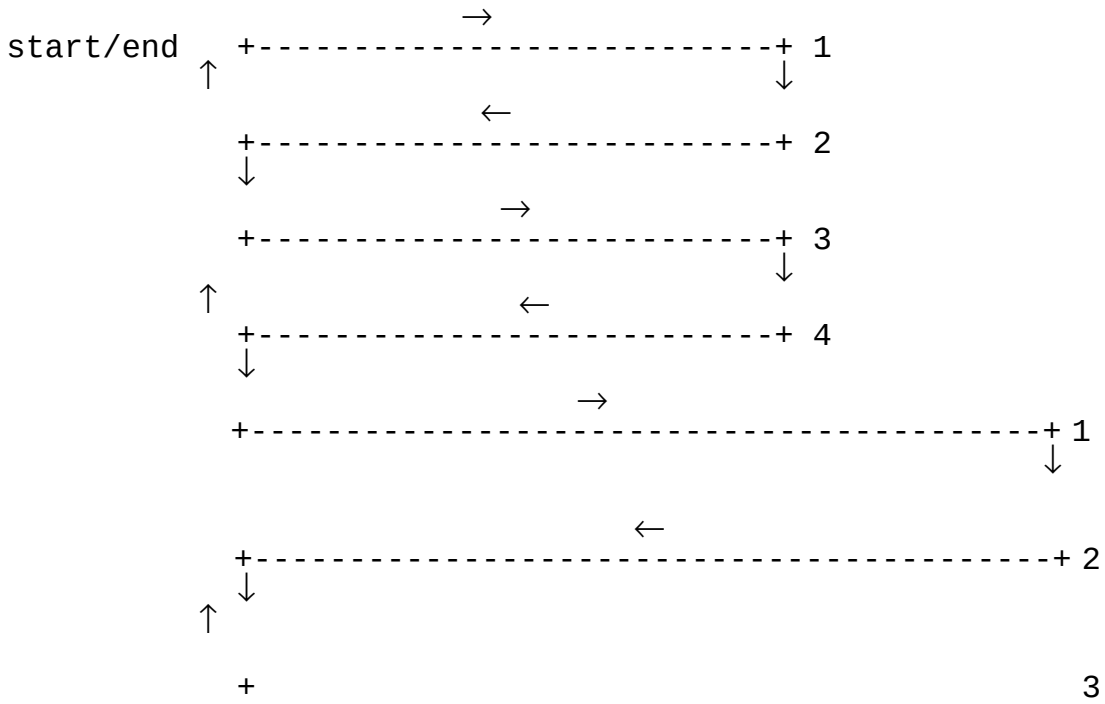
Example #10 RS232,Keyb	Motors run	RAM usage (bytes)	Function
X,Y Matrix	2	22	Mirror-image X,Y Matrix

P3,I1M-400,L3,I2M400,L-4,I1M1600,LM-3<cr>



Example #11 RS232,Keyb	Motors run	RAM usage (bytes)	Function
Two Raster Scans	2	27	Two Different Raster Scans using Loop-to-marker

I1M2000,I2M300,L-4,LM0,I2M600,I1M3000,L-3,IA2M0<cr>

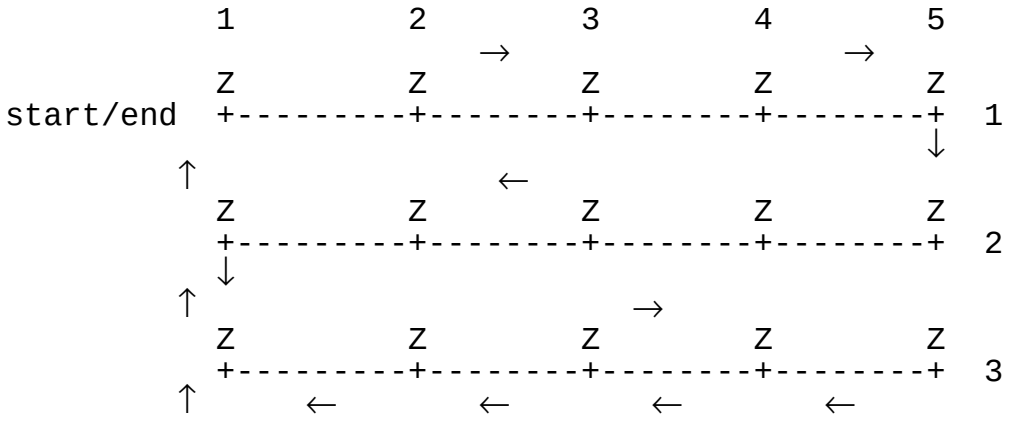


This would do the entire pattern of the previous example 5 times:

I1M2000,I2M300,L-4,LM0,I2M600,I1M3000<cr>
L-3,IA2M0,LM-0,LA5<cr>

Example #12 RS232,Keyb	Motors run	RAM usage (bytes)	Function
X,Y Matrix	3	30	X,Y Matrix Moving Z Axis Up then Down at each Position

I3M2000,I3M-2000,I1M1600,L5,I2M400,L-3,IA1M0,IA2M0<cr>



RS-232 Interactive Mode

The VP9000 can be controlled in an interactive mode. Interactive mode is when a host computer sends only the commands necessary to perform a single operation (usually an Index), then the host will wait for the VP9000 to finish before sending any additional commands. The following procedure would be used for running the VP9000 in an interactive mode:

1. The host puts the VP9000 On-Line by sending an "**F**"
2. The host sends a "**N**" to zero position registers if necessary
3. The host sends speed, and acceleration if necessary
4. The host sends an Index ("**I**" command, p.25 & p.26)
5. The host sends a "**R**" to start the Index
6. The host then will wait until it receives a ready prompt ("^") from the VP9000
   NOTE: The VP9000 does not send a carriage return or line feed following the "^", refer to the "**G**" command on p.44 to use instead of the "**F**" command in step 1
7. The user's routine for outputting, measuring, etc. would be executed by the host
8. A "**C**" would be sent from the host to clear the previous Index command from the VP9000's memory
9. The process is repeated from step # 3

Below is an interactive example written in BASIC: (This program is included on the Utility Disk under the name "VPEXINT")

```
95 REM Open RS-232 (COM1:), 9600 Baud, control lines disabled,ASCII
100 OPEN "COM1:9600,E,7,2,CS0,DS0" FOR RANDOM AS #1
105 REM Enable with echo off, Zero position registers
110 PRINT #1, "FN"
115 REM Clear any existing program, 1000 Steps/Sec,Index 1500 steps
120 PRINT #1, "C S1M1000,I1M1500,R"
130 GOSUB 500
135 REM Clear existing program, Index 900 steps
140 PRINT #1, "C I1M900,R"
150 GOSUB 500
155 REM Clear existing program, Index 1000 steps
160 PRINT #1, "C I1M1000,R"
170 GOSUB 500
175 REM Clear existing program, 2000 Steps/Sec, Index to zero
180 PRINT #1, "C S1M2000,IA1M0,R"
190 GOSUB 500
390 PRINT "DONE RUNNING VP9000"
392 REM leave VP9000 off-line
395 PRINT #1, "Q"
400 END
499 REM Wait until ready ("^") prompt appears in receive buffer
500 C$ = INPUT$(1, #1)
510 IF C$ <> "^" THEN 500
512 REM NOTE: Lines 515-540 are optional, they demonstrate two-way communication
515 REM Request motor position from VP9000 and print on your computer
520 PRINT #1, "X"
530 INPUT #1, P
540 PRINT "MOTOR POSITION="; P
550 REM Your routine for end of Index would go here
600 RETURN
```

Look at the VP9000 software diskette for other examples written in QuickBASIC, C, and PASCAL

RS-232 Download/ Stand-alone Mode

Entire programs can be transferred to the VP9000 over the RS-232 interface. When a program(s) has been downloaded to the VP9000, the VP9000 will keep the program(s) in memory until a Clear command is used. The ability to retain programs allows the VP9000 to be used in a stand-alone mode. In a stand-alone mode the operator starts the program by selecting "Run" from the Program Submenu, or by using the Run (I/O,4) input on the I/O connector (see "Appendix C" (p.59) for proper remote Run input connections).

Below is a BASIC example that sends example #6 as a stand-alone program: (This program is included on the Utility Disk under the name "VPEXSTD")

```
95 REM Open RS-232 (COM1:), 9600 Baud, control lines disabled,ASCII
100 OPEN "COM1:9600,E,7,2,CS0,DS0" FOR RANDOM AS #1
105 REM Enable with echo off, Zero position registers, select & clear program #1
110 PRINT #1, "FN PM-1,P10,I1M400,L10,L-2,"
999 END
```

Another way of sending programs from an IBM PC would be to first write them as text files with an editor such as DOS EDLIN, DOS Editor, Windows Notepad, Xtree, or Wordstar nondocument, and use the DOS commands MODE and COPY to send them to the VP9000. Microsoft Windows Terminal can send text files directly out the COM ports.

If the following program was stored under the file name of "EXAMPLE":

FN PM-1,P10,I1M400,L10,L-2,

Then the following DOS commands would send the program at 9600 baud to a VP9000 connected to COM1 serial port of the PC:

```
MODE COM1:9600,E,7,2
COPY EXAMPLE COM1
```

Daisy-chaining VP9000 Controllers

Users that require more than four motors or simultaneous motion can daisy-chain VP9000 Controllers together. Multiple Controllers can be operated from a single RS-232 port by daisy-chaining. When daisy-chaining multiple VP9000s together, addressing is accomplished by enclosing information in braces. Information between braces is relayed to succeeding Controllers. The number of braces used determines the destination of the information. As each Controller receives the information, it removes a set of braces and relays the remainder of the instruction to the next Controller. Up to 255 Controllers can be daisy-chained together and programmed in this manner. For example, if four Controllers are daisy-chained together and the fourth Control is to be programmed, the instruction for that Controller would be placed within three braces. The first Controller receiving the information would remove one set of braces and relay the information along, the second Controller would remove the second set of braces and the third Controller would remove the third set and relay the instructions to the fourth without any braces. The targeted fourth Controller would then be programmed with the information.

Examples:

This character will put all the VP9000s On-Line.

&

This will take two VP9000s linked together off-line.

{Q}Q

This will program Controller #4 to Index Motor 1 800 Steps.

{{{C,I1M800,R}}}

This polls Controller #2 to see if it is busy.

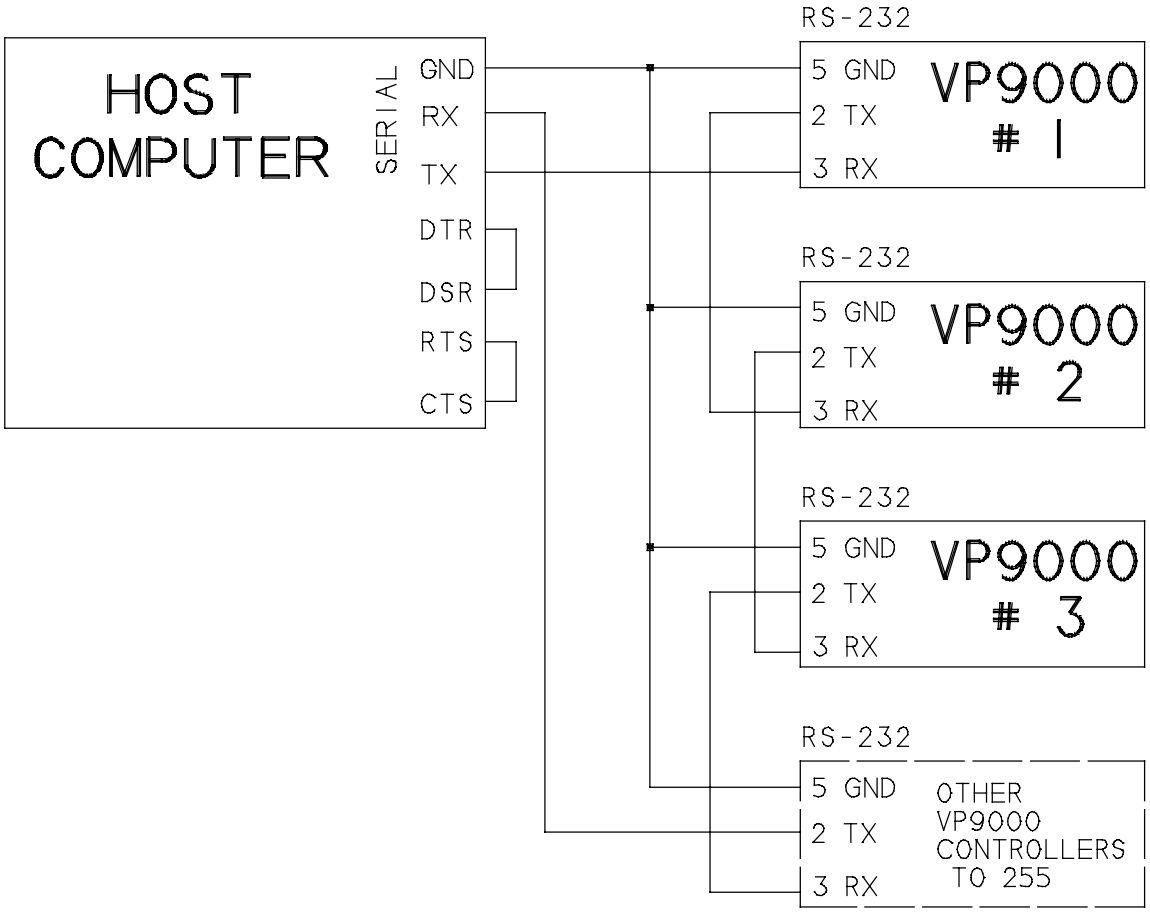
{V}

This requests position of Motor 1 of Controller #3.

{{X}}

The procedure for daisy-chaining Controllers is as follows:

1. Connect the RS-232 from the host to the VP9000 Controllers as shown.



2. Set RS-232 parameters on the host and Controllers the same.

3. Initialize Controllers by sending the following character (**CAUTION**: do not use the "**E**" or "**F**" command to put the VP9000s On-Line):

&

The "**&**" commands all the Controllers On-Line with echo off (the host will receive a "**!**" from the last VP9000).

4. Controller #1 is now ready to receive commands, all other controllers will be in a relay mode. Use braces to address a different VP9000 in the chain.

Reference

This section gives detailed explanations of the VP9000's commands.

Most of the commands use the VP9000's (NVRAM) memory. The required memory needed per command is specified in the following descriptions. The VP9000 has 256 bytes of NVRAM per program available for command storage. Each program can be cleared by a "**C**" or "**PM-x**" command. Additionally, all programs and all setup parameters can be reset by selecting "Defaults" from the Setup Submenu, refer to "Appendix A" (p.51). By utilizing the "**Jx**" (Jump to program **x**) command it is possible to link multiple programs together into one very long program. A more modular programming style can be utilized by using the "**JMx**" (Jump to program **x** and come back) command. By having a main program that Jumps to other programs (modules) and returns, can make long programs easier to maintain and edit. For example, maybe a module program #1 would be just for homing each axis to a limit switch, program #2 could initially set all motor speeds and accelerations, and program #3 might contain all the moves. Program #0 could be the main program that jumps to each of the module programs 1,2, and 3.

The Difference Between Incremental and Absolute Indexes

An incremental Index is, a move relative to the present position, a distance and direction specified by the Index from the present position.
An absolute Index is, a move relative to absolute zero position, a distance and direction from the present position calculated by the VP9000 based on absolute zero position.
Absolute zero is established by use of the "**N**", "**IA***m***M-0**" command, or "0" key in the Jog/slew Mode.

Calculating the Number of Steps For Indexing

To convert from "real" units to steps, divide the distance desired to move by the Adv/step. **Example:** To move 2 inches with the P2.5/W4 lead screw (2 ÷ 0.001 = 2000) requires a 2000 step index.

UniSlide Lead Screw	Adv/rev	Adv/step	Speed at 1000 steps/sec
P40,C	0.025"	0.0000625"	0.0625 ips
P20,B	0.050"	0.000125"	0.1250 ips
P10,W1	0.100"	0.00025"	0.2500 ips
P5,W2	0.200"	0.0005"	0.5000 ips
P2.5,W4	0.400"	0.001"	1.0000 ips
WF	1.000"	0.0025"	2.5000 ips
K1,Q1	1.00 mm	0.0025 mm	2.50 mm/sec
K2,Q2	2.00 mm	0.005 mm	5.00 mm/sec
K4,Q4	4.00 mm	0.01 mm	10.00 mm/sec

UniSlide Rotary Table	Gear Ratio	Adv/step	Speed at 1000 steps/sec
B4872TS	72:1	0.0125°	12.5°/sec
B4836TS	36:1	0.025°	25.0°/sec
B4818TS	18:1	0.050°	50.0°/sec
	180:1	0.005°	5.0°/sec
B5990TS	90:1	0.010°	10.0°/sec
B5945TS	45:1	0.020°	20.0°/sec

Commands For Front Panel and RS-232 Programming

Commands From RS-232 Interface:
When sending commands that require a value, the commands must end with a carriage return (Enter key or Return on most keyboards), comma, or a period.
Commands Entered on the VP9000 Front Panel:
Commands that require a value must end with a "Ent" selected from the display Menu.

I*m***M***x* Set steps to incremental **I**ndex (move) motor CW (positive, UniSlide Slider will move away from motor end , UniSlide Rotary Table will rotate CCW), *m*= motor# (1,2,3,4), *x*=1 to 16,777,215.
Memory usage = **4 bytes**.
Examples: **NOTE:** The "<cr>" is a carriage return character (<Enter> key on most keyboards). Command characters are in **LARGE BOLD**.

This example sets motor 1 to index 1200 steps CW:
I1M1200<cr>
This example sets motor 2 to index 9200 steps CW:
I2M9200<cr>
This example sets motor 3 to index 10200 steps CW:
I3M10200<cr>

I*m***M-***x* Set steps to incremental **I**ndex (move) motor CCW (negative, UniSlide Slider will move toward motor end , UniSlide Rotary Table will rotate CW), *m*= motor# (1,2,3,4), *x*=1 to 16,777,215.
Memory usage = **4 bytes**.
Examples:
This example sets motor 1 to index 120 steps CCW:
I1M-120<cr>
This example sets motor 2 to index 20 steps CCW:
I2M-20<cr>
This example sets motor 4 to index 1 step CCW:
I4M-1<cr>

IA*m***M***x* Set an **I**ndex to an **A**bsolute position, the distance and direction for the move from the present position is calculated by the VP9000 based on absolute zero position, *m*= motor# (1,2,3,4), *x*= ±1 to ±16,777,215. **NOTE:** Since the absolute position registers have a range of -8,388,608 to 8,388,607 steps, *x* should not be set to any number less than -8,388,608 or greater than 8,388,607.
Memory usage = **4 bytes**.
Examples:
This example sets motor 1 to index to absolute position 1200 :
IA1M1200<cr>
This example sets motor 4 to index to absolute position -90200 :
IA4M-90200<cr>

IA*m*M0 Set an **I**ndex to **A**bsolute zero position, *m*=motor# (1,2,3,4). When this command is used the VP9000 calculates the distance and direction to get back to absolute zero position. The "absolute zero" position was established when the "**N**" (Null Absolute Position Registers), "**IA*m*M-0**" command was used, or when the "0" key was pressed in the Jog/slew Mode.
Memory usage = **4 bytes**.
Examples:
This example sets motor 1 to index to absolute zero position:

IA1M0<cr>

This example sets motor 2 to index to absolute zero position:

IA2M0<cr>

This example sets motor 3 to index to absolute zero position:

IA3M0<cr>

IA*m*M-0 Zero motor position for motor# *m*, *m*=motor# (1,2,3,4). This command clears the position register for the motor selected, making this position absolute zero. The display will show all zeros for the motor selected.
Memory usage = **4 bytes**.
Examples:
This example makes the present position for motor 1 absolute zero:

IA1M-0<cr>

This example makes the present position for motor 2 absolute zero:

IA2M-0<cr>

I*m*M0 Move positive until the positive limit switch is encountered (Home to Positive Limit), *m*=motor# (1,2,3,4). If the limit switch input was disabled from the Setup menu ("Appendix A," p.51) the limit switch input will be re-enabled for the duration of this command. The Index will end if the limit switch is not encountered after 16 million steps.
Memory usage = **4 bytes**.
Example: This example sets motor 1 to seek the positive limit switch:

I1M0<cr>

I*m*M-0 Move negative until the negative limit switch is encountered (Home to Negative Limit), *m*=motor# (1,2,3,4). If the limit switch input was disabled from the Setup menu ("Appendix A," p.51) the limit switch input will be re-enabled for the duration of this command. The Index will end if the limit switch is not encountered after 16 million steps.
Memory usage = **4 bytes**.
Example: This example sets motor 1 to seek the negative limit switch:

I1M-0<cr>

S*m***M***x* Set **S**peed of motor (70% power applied to motor), *m*= motor# (1,2,3,4), *x*=0 to 8000 steps/sec. in 1 step/sec. intervals, 0= 1 step/ 2 sec. If this command is never used, the default speed will be 2000 steps/sec. The default speed is updated to the set speed every time the speed command is used. A "**C**" (Clear) or **PM**-*x* command does not alter the default speed. The VP9000 will limit the speed to a maximum of 8000 if a value higher is entered.

CAUTION: motor torque decreases as speed increases, and some motors have limited torque above 2000 steps/sec. If the motor torque is below the needed torque to move the load, the motor will stall (lose synchronism and proper position), see Appendix F for speed/torque curves. Stepping motors produce vibration at certain speeds, if noise or vibration at a chosen speed is objectionable, try a speed slightly higher or lower.

Memory usage = **3 bytes**.

Example:

This example sets the speed of motor 1 to 500 steps/sec at 70% power:

S1M500<cr>

When the "**S**" speed command is used for setting speed, motor running torque will be 70% of the torque shown in "Appendix F". For most applications 70% motor torque will be more than adequate. For moving heavy loads and to maximize speed, the "**SA**" speed command (100% power) may be needed.

NOTE: Motor power will always be zero when the motor is stationary (motors are normally unenergized at a standstill, Motor 1 has settable holding torque which is a fixed amount of electrical current).

Advantages of the "S" speed command (70% motor power/torque)

1. Saves 45 watts of energy when motor is operating.
2. Motors run smoother and quieter at low and mid-range speeds.
3. Suppresses the natural motor resonance (torque loss) common in the 140 to 250 step per second range, and at mid-range speeds on some motors.
4. Reduces Motor and Controller heating.

SA*m***M***x* Set **S**peed of motor (100% power applied to motor), *m*= motor# (1,2,3,4), *x*=0 to 8000 steps/sec. in 1 step/sec. intervals, 0= 1 step/ 2 sec.

Example:

This example sets the speed of motor 2 to 3000 steps/sec at 100% power:

SA2M3000<cr>

CAUTION: Motor and Controller surface temperatures become hot when running motors continuously. Only use 100% ("**SA**" command) motor power if maximum torque is required. For maximum efficiency when lifting heavy loads vertically, use the "**SA**" command to set speed for traversing upwards, and use the "**S**" speed command for the speed down.

A*m***M***x* **A**cceleration / deceleration, *m*= motor# (1,2,3,4), *x*=1 to 127. A value of 1 is 1000 steps / sec^2, 2 is 2000 steps / sec^2, 127 is 127000 steps / sec^2, etc. If this command is never used, the default will be 2 (2000 steps / sec^2). The default is updated to the set acceleration / deceleration every time the command is used. A "**C**" (Clear) or **PM**-*x* command does not alter the default. The higher the number used, the faster the motor will reach the set speed, and the faster it will slow down to a stop. **CAUTION**: motors may stall if this value is set to high.
Memory usage = **2 bytes**
Example:
This example sets the acceleration / deceleration of motor 1 to 3000 steps / sec^2:

A1M3<cr>

How to Determine Maximum Speed

1. Set acceleration to a value of 2. If the traverse is very short, it may not be possible to reach full speed before running out of travel. If this is the case, the acceleration will have to be set to a higher value as the speed is increased.
2. Increase speed (use "**SA**" command) in 1000 step / sec. increments until the motor stalls.
3. Reduce speed in 500 step / sec. intervals until the motor runs without stalling.
4. Refer to the graphs on the last page of this addendum to find the torque for the motor at the present speed.
5. Multiply the torque by 1.2, and find the speed corresponding to this torque from the graph, use this speed as a maximum setting.
   NOTE: use 1.2 (+20%) multiplier as a minimum margin, a better safety factor would be 1.4 (+40%).

How to Determine Maximum Acceleration

1. Set speed to maximum as determine from the previous procedure.
2. Increase acceleration by increments of 5 until the motor stalls.
3. Decrease acceleration until motor does not stall.
4. Multiply the present acceleration number by 0.8 to determine the maximum acceleration (round to the nearest integer).
   NOTE: For added safety, multiply by 0.6 instead of 0.8.

Programming Shortcut:
The motor designation in Acceleration, Speed, and Index commands is optional if the desired motor has already been set as the current motor. The current motor is motor 1 when the Controller is first turned on. The last motor jog / slewed will be the current motor number. The current motor will be the number used in the last Acceleration, Speed, or Index command. Users that have only a one motor VP9000 (Model VP9001) do not have to use the motor designation in a command. For example, these commands would always be motor 1 commands of a one motor VP9000:

A2,S4000,I400,

For running a particular motor of a multi-motor VP9000, only the first Command needs the motor number. For example, all of these commands would be for motor 2:

I2M200,I-200,S2000,IA0,

L0 Loop continually from the beginning or Loop-to-marker of the current program. The loop will occur to the last Loop-to-marker of the current program if it was set previously. This command can be used once in a program as the last command, it functions the same as a "continuous run input".
Memory usage = **1 byte**.

LM0 Sets the **L**oop-to-marker at this point in the current program. All looping commands in the current program that follow will branch to here. Any loop commands in the program prior to this marker will branch to the beginning of the program or a previous marker.
NOTE: Multiple markers can be used in a program, the number is only limited by the program memory available (256 bytes per program).
Memory usage = **1 byte**

LM-0 Resets the **L**oop-to-marker to the beginning of the current program.
NOTE: Multiple resets can be used in a program, the number is only limited by the program memory available (256 bytes per program).
Memory usage = **1 byte**

L*x* **L**oop from beginning or Loop-to-marker of the current program *x*-1 times (*x*=2 to 65,535). Loop commands can be nested up to 33 active at one time.
NOTE: When the Loop reaches its last count, the non-loop command directly preceding the Loop will be ignored.
Memory usage = **3 bytes**.

Example:
This example sets a loop to repeat, any previous commands 4000-1 times, while repeating the directly preceding non-loop command 4000-2 times:

L4000<cr>

L-*x* **L**oop from beginning or Loop-to-marker of the current program *x*-1 times alternating direction of motor 1 indexes (*x*=2 to 65,535). Loop commands can be nested up to 33 active at one time.
NOTE: When the Loop reaches its last count, the non-loop command directly preceding the Loop will be ignored.
Memory usage = **3 bytes**.

Example:
This example sets a loop to repeat, any previous commands 100-1 times alternating motor 1 direction every repeat, while repeating the directly preceding non-loop command 100-2 times:

L-100<cr>

LA*x* **L**oop **A**lways from beginning or Loop-to-marker of the current program *x*-1 times (*x*=2 to 65,535). Loop commands can be nested up to 33 active at one time.
Memory usage = **3 bytes**.

Examples:
This example sets a loop to repeat all previous commands 4000-1 times:

LA4000<cr>

Consecutively nested loops are equal to the product of their loop values. For example, the following loops together are equal to 10,000,000-1 (50,000 x 200):

LA50000,LA200<cr>

LA-*x* **L**oop **A**lways from beginning or Loop-to-marker of the current program *x*-1 times alternating direction of motor 1 indexes (*x*=2 to 65,535). Loop commands can be nested up to 33 active at one time.
Memory usage = **3 bytes**.

Examples:
This example sets a loop to repeat 100-1 times all previous commands alternating motor 1 direction every repeat:

LA-100<cr>

Consecutively nested loops are equal to the product of their loop values. For example, the following loops together are equal to 2,500,000,000-1 (50,000 x 50,000):

LA-50000,LA50000<cr>

LM-2 **L**oop once from the beginning or Loop-to-marker of the current program, reversing index direction of motor 2. See "Example Programs" section (p.15) for use of this command.
Memory usage = **1 byte**.

LM-3 **L**oop once from beginning or Loop-to-marker of the current program, reversing index direction of motor 1 and motor 2. See "Example Programs" section (p.15) for use of this command.
Memory usage = **1 byte**.

P*x* **P**ause *x* tenths of a second (*x*=0 to 65,535). A "P" will appear at the lower right on the front panel display while pausing.
Memory usage = **3 bytes**.

Examples:
This example pauses for 1 second:

P10<cr>

This example pauses for 15 seconds:

P150<cr>

This example pauses for 1 hour:

P36000<cr>

P-*x* **P**ause *x* tenths of a millisecond (*x*=1 to 65,535).
Memory usage = **3 bytes**.

Example:
This example pauses for 1 millisecond (0.001 seconds):

P-10<cr>

PA*x* **P**ause *x* tenths of a second **A**ltering the state of output 1 (*x*=0 to 65,535, 10 µsec pause when *x*=0). The user output 1 (I/0,12) will go to +5V for the duration of the pause. A "P" will appear at the lower right on the front panel display while pausing.
Memory usage = **3 bytes**.

Example:
This example pauses for 15 seconds holding output 1 high:

PA150<cr>

PA-*x* **P**ause *x* tenths of a millisecond **A**ltering the state of output 1 (*x*=1 to 65,535). The user output 1 (I/0,12) will go to +5V for the duration of the pause.
Memory usage = **3 bytes**.

Example:
This example pauses for 15 milliseconds (0.015 seconds) holding output 1 high:

PA-150<cr>

PM*x* Select **P**rogra**m** number *x* as the current program, *x*= 0 to 30. Each program can hold 256 bytes of commands. If this command is never used, the default program number will be 0.
Programs 21 to 28 can be interactive with user inputs from the remote Jog controller, or the front panel keyboard, refer to **U12**, **U13**, **U22**, and **U23** commands for more information. Also, programs 29 and 30 interact with user inputs 3 and 4 (I/O,7 and I/O,8) refer to "Appendix C" (p.59) for more information.
Memory usage = **0 bytes**. This command is immediate (not stored)

Example:
This example selects program #10 for the current program:

PM10<cr>

PM-*x* Select and clear all commands from **P**rogra**m** number *x*, *x*= 0 to 30. This command will select program *x* as the current program and delete all commands from this program. Any speed, acceleration, or motor position settings (defaults) will not be altered, the state of user outputs, Backlash Compensation, Indicate Limit Switch Over-travel will not be affected,
Memory usage = **0 bytes**. This command is immediate (not stored)

Example:
This example selects program #0 and erases all commands within it:

PM-0<cr>

J*x* **J**ump to the beginning of program number *x*, *x*= 0 to 30. Program number *x* will temporarily be the current program, all commands will be executed starting from the first one that was previously entered into program *x*. If there is not any commands in program *x*, or after executing the last command, the program will end, and the VP9000 will send the ready prompt to the host ("^"). The current program number will still be the program that was originally selected with a "**PMx**" or "**PM-x**" command. Linking multiple programs (maximum of 31) together is possible by using a jump command, as the last command, to make a jump to a different program. Linking programs together makes it possible to make one or more programs much longer then the 256 byte per program memory limit. All looping commands in program *x* will be local to this program (all looping is specific to the current program, looping never occurs outside the current program).

Memory usage = **2 bytes**

Example:
This example will jump to program #10 :

J10<cr>

JM*x* Jump to the beginning of program number *x* and come back for **M**ore after program *x* ends, *x*= 0 to 30. Program number *x* will temporarily be the current program, all commands will be executed starting from the first one that was previously entered into program *x*. If there is not any commands in program *x*, or after executing the last command, control will be transferred back to the program that initiated the Jump, then the next command in the initiating program will be executed. It is possible to have up to 30 **JM***x* commands active at one time. This command can be used to make programming more modular, having a main program that jumps to other programs (modules) and returns, can make long programs easier to maintain and edit. All looping commands in program *x* will be local to this program (all looping is specific to the current program, looping never occurs outside the current program).
CAUTION: Motor reverse-direction-flags are set by "**L-x**", "**LM-2**", and "**LM-3**" looping commands. If a **JM***x* command is used "inside" one of these loops, (of the program initiating the **JM***x* command) motor 1 and motor 2 direction may be reversed in program *x*.
Memory usage = **2 bytes**

Example:
This example will jump to program #10 and return:

JM10<cr>

U0 Wait for a "high" on the user input 1. A "high" is a voltage between +3VDC and +25VDC applied to I/O,5. A simple push-button or toggle switch can be used between +5V (I/O,2) and input 1 (I/O,5) to satisfy this input. The input level must be high for at least 1 ms to be a valid input. A "U" will appear at the lower right on the front panel display while waiting. This command is best used when interfacing to other solid-state logic devices, refer to the "**U50**" command for push-button switch input.
Memory usage = **2 bytes**.

U1 Wait for a "high" on the user input 1, holding user output 1 "high" (+5V) while waiting. A "high" is a voltage between +3VDC and +25VDC applied to I/O,5. User output 1 (I/O,12) will go to +5V for the duration of the wait. The input level must be high for at least 1 ms to be a valid input. A "U" will appear at the lower right on the front panel display while waiting. This command is best used when interfacing to other solid-state logic devices, refer to the "**U51**" command for push-button switch input.
Memory usage = **2 bytes**.

U2 Enable Jog while waiting for an input. This command will allow jog/slewing, with the Remote Jog Controller, during one of the following wait commands: **U0,U1,U10,U11,U12,U16,U17,U22,U30,U31,U40,U41,U50,U51,U60**, or **U61**.
Memory usage = **2 bytes**.

U3 Disable Jog while waiting for an input (default setting on power-up). This command will disable jog/ slewing, with the Remote Jog Controller, during any wait command.
Memory usage **2 bytes**.

U4 User output 1 "low". The user output 1 (I/O,12) will go to 0V. This is the state of the user output 1 on power-up. This command is used in conjunction with the "**U5**" command.
Memory usage = **2 bytes**.

U5 User output 1 high. The user output 1 (I/O,12) will go to +5V. This command is used in conjunction with the "**U4**" command.
Memory usage = **2 bytes**.

U6 Send "W" to the host and wait for a "G" to continue. The VP9000 sends the single character "W" to the host when this command is executed. The VP9000 will wait until a "G" is received from the host before proceeding in the program. A "W" will appear at the lower right on the front panel display while waiting.
Memory usage = **2 bytes**.

U7 Start of Continuous Index with pulse output. This command is used when it is desirable to make several Indexes on one axis without stopping or slowing between each Index. Instead of stopping a 20 µsec wide positive going pulse will appear on user output 2 (I/O,13) at each Index distance. This pulse would be used to trigger measurement/sampling equipment. The "**U9**" command must be used as the last command to decelerate properly to a stop from the last Index.
Memory usage = **2 bytes**.

Continuous Indexes require the following:

a) Each Index must be the same motor, and direction.

b) After the last Index, the motor will move a deceleration distance which will be one half the distance to accelerate to full speed.

c) The maximum speed that should be used is 4000 steps/sec. Minimum speed is 31 steps/sec.

d) Up to six speeds changes are allowed. The VP9000 will accelerate or decelerate at the current Acceleration/deceleration value to the new speed without stopping. The first command after a U7 or U8 should not be a speed command. **Loop commands are not allowed if speed commands are used within the Continuous Index**.

e) No Acceleration, Pause, -Loop, Jump, or Wait commands are allowed in Continuous Indexes.

f) The "**D**" and the "**H**" commands are not useable during a continuous index.

U7 command examples:

This example makes an index on motor 1, producing a pulse at positions 1000,1100,1150,1250, and then runs motor 1 back to the start position:

S1M1500,U7,I1M1000,I1M100,I1M50,I1M100,U9,IA1M0<cr>

This example will Index motor 2 and pulse 100 times:

U7,I2M400,LA100,U9<cr>

This example will run continuously, producing a pulse on each revolution (400 steps) of motor 1:

U7,I1M400,L0<cr>

This example makes an index on motor 2, producing a pulse with speed changes between each index:

S2M1500,U7,I2M2000,S2M3000,I2M4000,S2M500,I2M800,U9, S2M3000,IA2M0<cr>

U8 Start of Continuous Index sending "@" to the host. This command is the same as the "**U7**" except the single character "@" is transmitted at each Index distance, instead of a pulse on the user output 2. Always use the highest baud rate possible (9600).
Memory usage = **2 bytes**.

U9 End of Continuous Index. This command is used, as the ending command of a Continuous Index, in conjunction with the "**U7**" or "**U8**" commands.
Memory usage = **2 bytes**.

U10 Wait for a "low" on the user input 2. A "low" is a voltage less than 0.8 VDC (not to be less than 0V) applied to I/O,6. A simple push-button or toggle switch can be used between Gnd (I/O,1) and input 2 (I/O,6) to satisfy this input. The input level must be low for at least 1 ms to be a valid input. A "U" will appear at the lower right on the front panel display while waiting. This command is best used when interfacing to other solid-state logic devices, refer to the "**U60**" command for push-button switch input.
Memory usage = **2 bytes**.

U11 Wait for a "low" on the user input 2, holding user output 2 "high" (+5V) while waiting. A "low" is a voltage less than 0.8 VDC (not to be less than 0V) applied to I/O,6. User output 2 (I/O,13) will go to +5V for the duration of the wait. The input level must be low for at least 1 ms to be a valid input. A "U" will appear at the lower right on the front panel display while waiting. This command is best used when interfacing to other solid-state logic devices, refer to the "**U61**" command for push-button switch input.
Memory usage = **2 bytes**.

U12 Wait for a key to be pressed on the front panel keyboard. This command allows user interaction, by initiating a jump to a specific program, or allowing the current program to proceed.
The ← key will cause a jump to program #21.
The → key will cause a jump to program #22.
The ↓ key will cause a jump to program #23.
The ↑ key will cause a jump to program #24.
The 0 or SEL key will cause the current program to continue to the next command.
A "K" will appear at the lower right on the front panel display while waiting.
Memory usage = **2 bytes**.

U13 Wait for a button on the Remote Jog Controller to be pressed. This command allows user interaction, by initiating a jump to a specific program, or allowing the current program to proceed.
The 1- button will cause a jump to program #25.
The 1+ button will cause a jump to program #26.
The 2- button will cause a jump to program #27.
The 2+ button will cause a jump to program #28.
The ▭ button will cause the current program to continue to the next command
A "R" will appear at the lower right on the front panel display while waiting.
Memory usage = **2 bytes**.

U14 User output 2 "low". The user output 2 (I/O,13) will go to 0V. This is the state of the user output 2 on power-up. This command is used in conjunction with the "**U15**" command.
Memory usage = **2 bytes**.

U15 User output 2 high. The user output 2 (I/O,13) will go to +5V. This command is used in conjunction with the "**U14**" command.
Memory usage = **2 bytes**.

U16 Wait for the ← key to be pressed on the front panel keyboard.
A ← will appear at the lower right on the front panel display while waiting.
Memory usage = **2 bytes**.

U17 Wait for the → key to be pressed on the front panel keyboard.
A → will appear at the lower right on the front panel display while waiting.
Memory usage = **2 bytes**.

U22 Wait for a key to be pressed on the front panel keyboard. This command allows user interaction, by initiating a jump-and-come-back-for-more to a specific program, or allowing the current program to proceed.
The ← key will cause a jump to program #21 and return.
The → key will cause a jump to program #22 and return.
The ↓ key will cause a jump to program #23 and return.
The ↑ key will cause a jump to program #24 and return.
The 0 or SEL key will cause the current program to continue to the next command.
A "K" will appear at the lower right on the front panel display while waiting.
Memory usage = **2 bytes**.

U23 Wait for a button on the Remote Jog Controller to be pressed. This command allows user interaction, by initiating a jump-and-come-back-for-more to a specific program, or allowing the current program to proceed.
The 1- button will cause a jump to program #25 and return.
The 1+ button will cause a jump to program #26 and return.
The 2- button will cause a jump to program #27 and return.
The 2+ button will cause a jump to program #28 and return.
The ▭ button will cause the current program to continue to the next command
A "R" will appear at the lower right on the front panel display while waiting.
Memory usage = **2 bytes**.

U30 Wait for a high to low transition on the user input 1. A "high" is a voltage between +3VDC and +25VDC applied to I/O,5. A simple pushbutton or toggle switch can be used between +5V (I/O,2) and input 1 (I/O,5) to satisfy this input. The input level must be high for at least 1 ms and go low (less than 0.8V) to be a valid input. A "U" will appear at the lower right on the front panel display while waiting. This command is best used when interfacing to other solid-state logic devices, refer to the "**U50**" command for push-button switch input.
Memory usage = **2 bytes**.

U31 Wait for a high to low transition on the user input 1, holding user output 1 "high" (+5V) while waiting. A "high" is a voltage between +3VDC and +25VDC applied to I/O,5. User output 1 (I/O,12) will go to +5V for the duration of the wait. The input level must be high for at least 1 ms and go low (less than 0.8V) to be a valid input. A "U" will appear at the lower right on the front panel display while waiting. This command is best used when interfacing to other solid-state logic devices, refer to the "**U51**" command for push-button switch input.
Memory usage = **2 bytes**.

U32 Wait for the 1- button to be pressed on the Remote Jog Controller. A "<" will appear at the lower right on the front panel display while waiting. Memory usage = **2 bytes**.

U33 Wait for the 1+ button to be pressed on the Remote Jog Controller. A ">" will appear at the lower right on the front panel display while waiting. Memory usage = **2 bytes**.

U40 Wait for a low to high transition on the user input 2. A "low" is a voltage less than 0.8 VDC (not to be less than 0V) applied to I/O,6. A simple push-button or toggle switch can be used between Gnd (I/O,1) and input 2 (I/O,6) to satisfy this input. The input level must be low for at least 1 ms and go high (+5V) to be a valid input. A "U" will appear at the lower right on the front panel display while waiting. This command is best used when interfacing to other solid-state logic devices, refer to the "**U60**" command for push-button switch input.
Memory usage = **2 bytes**.

U41 Wait for a low to high transition on the user input 2, holding user output 2 "high" (+5V) while waiting. A "low" is a voltage less than 0.8 VDC (not to be less than 0V) applied to I/O,6. User output 2 (I/O,13) will go to +5V for the duration of the wait. The input level must be low for at least 1 ms and go high (+5V) to be a valid input. A "U" will appear at the lower right on the front panel display while waiting. This command is best used when interfacing to other solid-state logic devices, refer to the "**U61**" command for push-button switch input.
Memory usage = **2 bytes**.

U50 Wait for a high to low transition on the user input 1 with debouncing for a mechanical push-button switch. A "high" is a voltage between +3VDC and +25VDC applied to I/O,5. A push-button or toggle switch would be connected between +5V (I/O,2) and input 1 (I/O,5) to satisfy this input. A "U" will appear at the lower right on the front panel display while waiting.

When a push-button switch is pressed, the switch's electrical contacts will bounce off each other a few times before settling into their final position. This bouncing will produce a series of highs and lows, which could result in several consecutive wait commands to see these electrical bounces as valid inputs from just one push-button press. When using the "U50" command, the VP9000 will filter out the electrical bounces associated with mechanical switches.
Memory usage = **2 bytes**.

U51 Wait for a high to low transition on the user input 1 with debouncing for a mechanical push-button switch, holding user output 1 "high" (+5V) while waiting. A "high" is a voltage between +3VDC and +25VDC applied to I/O,5. User output 1 (I/O,12) will go to +5V for the duration of the wait. A push-button or toggle switch would be connected between +5V (I/O,2) and input 1 (I/O,5) to satisfy this input. A "U" will appear at the lower right on the front panel display while waiting.

When a push-button switch is pressed, the switch's electrical contacts will bounce off each other a few times before settling into their final position. This bouncing will produce a series of highs and lows, which could result in several consecutive wait commands to see these electrical bounces as valid inputs from just one push-button press. When using the "**U51**" command, the VP9000 will filter out the electrical bounces associated with mechanical switches.
Memory usage = **2 bytes**.

U60 Wait for a low to high transition on the user input 2 with debouncing for a mechanical push-button switch. A "low" is a voltage less than 0.8 VDC (not to be less than 0V) applied to I/O,6. A push-button or toggle switch would be connected between Gnd (I/O,1) and input 2 (I/O,6) to satisfy this input. A "U" will appear at the lower right on the front panel display while waiting.

When a push-button switch is pressed, the switch's electrical contacts will bounce off each other a few times before settling into their final position. This bouncing will produce a series of highs and lows, which could result in several consecutive wait commands to see these electrical bounces as valid inputs from just one push-button press. When using the "**U60**" command, the VP9000 will filter out the electrical bounces associated with mechanical switches.
Memory usage = **2 bytes**.

U61 Wait for a low to high transition on the user input 2 with debouncing for a mechanical push-button switch, holding user output 2 "high" (+5V) while waiting. A "low" is a voltage less than 0.8 VDC (not to be less than 0V) applied to I/O,6. User output 2 (I/O,13) will go to +5V for the duration of the wait. A push-button or toggle switch would be connected between Gnd (I/O,1) and input 2 (I/O,6) to satisfy this input. A "U" will appear at the lower right on the front panel display while waiting.

When a push-button switch is pressed, the switch's electrical contacts will bounce off each other a few times before settling into their final position. This bouncing will produce a series of highs and lows, which could result in several consecutive wait commands to see these electrical bounces as valid inputs from just one push-button press. When using the "**U60**" command, the VP9000 will filter out the electrical bounces associated with mechanical switches.
Memory usage = **2 bytes**.

Commands For User Programmable Counter

The following "U" commands are for the user programmable counter that can total up to 4,294,967,295 counts. The count value is stored in the VP9000's NVRAM to maintain count total when power is turned off.

U70 Zero counter total. This command is used to reset the user counter to zero. **NOTE:** Remember, this command is stored in a program, and every time a "Run" occurs the counter will be zeroed.
Memory usage = **2 bytes**.

U71 Display counter total on the front panel display. This command will make the front panel display the dedicated readout for the user counter. The standard motor position readout will be replaced by the count total when this command is used. When the program containing this command is "cleared" ("**PM-x**" or "**C**" command) the display will automatically be set back to the standard motor position readout. See "**U72**" command to turn off counter display.
Memory usage = **2 bytes**.

The counter display will look like this:

```
User Programable Counter
COUNTS=   0000000000
```

U72 Turn off the counter total on the front panel display. This command will set the display back to the standard motor position readout from being the user counter readout. See "**U71**" command to turn on counter display.
memory usage = **2 bytes**.

U73 Increment the counter by one. This command will add one to the user counter total, and if the counter display is on, the new total will be updated to the display. The maximum value the counter can hold is 4,294,967,295 counts. The count total will be maintained even if power is turned off or disconnected from the VP9000.
Memory usage = **2 bytes**.

Example: Life testing push-button switches; the **U0** is used as a continuity check on a normally-open switch connected between input 1 and +5V.
This example will: a) Enable the counter display.
b) Run motor 1 to actuate the switch.
c) Wait for input 1 (continuity check).
d) Increment counter by one and repeat from step "b".

U71,LM0,I1M400,U0,I1M-400,U73,L0<cr>

When the switch fails, the program will stop at the **U0** command with the number of cycles to failure being displayed on the front panel.

U90 Wait for a high to low transition on the Run input with debouncing for a mechanical push-button switch. A "high" is a voltage between +3VDC and +25VDC applied to I/O,4. A push-button or toggle switch would be connected between +5V (I/O,2) and Run input (I/O,4) to satisfy this input. A "R" will appear at the lower right on the front panel display while waiting. **CAUTION:** The Run input also starts the current program when the VP9000 is in an idle state either On-Line or in Jog/slew mode, refer to "Appendix C" (p.59).
Memory usage = **2 bytes**.

Commands Specific For RS-232 Interfacing

All of the previous commands listed are applicable to programming over the RS-232 interface and from the VP9000's front panel keyboard. Additionally, the following commands are only applicable to programming from a host over the RS-232 interface, they are ended with a carriage return (Enter key), comma, or period:

B*x* **B**acklash Compensation, compensation is on when *x*=1, off when *x*=0 (default). **NOTE:** Backlash Compensation can also be set from the front panel, refer to "Appendix A" (p.51) for setting from the Setup menu.
The VP9000 can compensate for mechanical backlash by ending every index in the positive direction. When backlash compensation is on, and a motor makes a negative Index, 20 steps will be added to the Index. The Motor will then immediately reverse, indexing positive 20 steps.
Memory usage = **0 bytes** (this command is immediate, and uses a reserved NVRAM memory location)

O*x* Indicate Limit Switch **O**ver-travel to the host, off when *x*=0 (default on power-up), the VP9000 sends "O" when *x*=1 and a limit switch is encountered. This command is useful when the host needs to know if a positioner's travel has been exceeded due to a motor stall or an index(es) that are too long. When Indicate-Limit-Switch-Over-travel is on, the VP9000 transmits the single character "O" to the host when an indexing motor activates it's limit switch input.
Memory usage = **0 bytes** (this command is immediate, and uses a reserved memory location)

NOTE: limit switches also stop motor motion immediately.

PM Request the number of the current **P**rogra**m**. The VP9000 will send the current program number (00 to 30) followed by <cr> (carriage return). The program number is determined by previously using the **PMx** or **PM-x** command.
Memory usage = **0 bytes** (this command is immediate, and does not use any of the VP9000's memory)

The following are immediate (not stored) commands, therefore they do not use any of the VP9000's memory and do not need an ending carriage return, comma, or period:

Q **Q**uit On-Line mode (return to Jog/slew mode). The "**Q**" command is used to get back to the power-up state, where the VP9000 is in the Jog/slew mode the (On-Line light is off).

R **R**un currently selected program. The "**R**" command will start execution of commands stored (of current program) in the VP9000's memory. At the end of the "run" the single character "^" (no carriage return or line feed follows the "^") will be transmitted to the host. Additional "**R**" commands received by the VP9000 will repeat the same program. See the "**C**" and "**PMx**" command to clear a program from memory. The Run input (I/O,4) functions the same as this "**R**" command.

N **N**ull (zero) motors 1,2,3,4 Absolute Position Registers. This command can be used in the Jog/slew or the On-Line mode. The "**N**" command zeros the position registers that have been counting steps from indexing and/or jog/slewing the motor(s).
NOTE: The Display will be "blanked-out" until the next Index command is executed.

K **K**ill operation in progress. This command will immediately interrupt any running program. The user outputs will be reset, all looping and hold flags will be reset, and if a motor is indexing it will be stopped immediately. If the motor speed is above 1000 steps/sec. when the interrupt occurs, the motor may loose position due to mechanical overshoot (see the "**D**" command for a less abrupt method to interrupt indexes). The VP9000 will transmit the "^" to the host after receiving the "K" command.

V **V**erify Controller's status, when On-Line the VP9000 sends a "B" to the host if it is busy, or an "R" if it is ready. The "**V**" command is used to poll the VP9000 to see if it is busy running a program, or ready to receive more commands. Use of this command is optional, since the VP9000 automatically transmits a "^" character to the host when a program has finished. If the VP9000 is running a program when it receives a "**V**" the VP9000 will respond by transmitting the single character "B". If the VP9000 is idle waiting for a command the VP9000 will respond by transmitting the single character "R". When in the Jog/slew mode the VP9000 will always respond by sending a "J".

C **C**lear all commands from the currently selected program. Any speed, acceleration, or motor position settings (defaults) will not be altered, the state of user output, Backlash Compensation, Indicate Limit Switch Over-travel will not be affected.

D **D**ecelerate to a stop (interrupts current index in progress). When the VP9000 receives the single character "D" while it is indexing a motor, that motor will be decelerated to a stop at the set deceleration. The motor position prior to decelerating is saved, refer to the "*" command to request this position. The VP9000 will then proceed to the next command in the program. The "**D**" command has a different function when in the Jog/slew mode, refer to the section on "Digitizing With a Host" (p.6) for more information.

***** (Asterisk) Request motor position when the last "**D**" command or input on User input 4 occurred. When the VP9000 receives the single character "*****" it will transmit the motor position saved at the time an index was interrupted. The interrupted index would have been initiated by a "**D**" (Decelerate to a stop) command or a high to low transition on User input 4 (Pin #8 of I/O). Refer to the "**D**" and I/O connections for more information on interrupting motor indexes. Below is what the host would receive if the last motor indexing was interrupted at position negative 14901.

-0014901<cr>

E **E**nable On-Line mode with echo on. The single character "**E**" is used to put the VP9000 in the On-Line mode after power-up. All characters the VP9000 receives will be echoed back to the host. Refer to the section "On-Line Mode/ Programming from a Host" (p.13) for more information.

F Enable On-Line mode with echo o**FF**. The single character "**F**" is used to put the VP9000 in the On-Line mode after power-up. No characters will be echoed back to the host. The VP9000 will still respond to motor position and status requests. Refer to the section "On-Line Mode/ Programming from a Host" (p.13) for more information.

G Enable On-Line mode with echo off, **G**rouping a <cr> (carriage return) with the "^", "**<**", "**>**", "**=**", "**:**", "**W**", "**O**" character responses.

This command has the functionality of the "**F**" command except a <cr> will be appended to: The "^" (Ready prompt) after finishing a program; the "**R**" response to a "**V**" command; an "**O**" response to Limit Switch Over-travel; the "**:**" response to a "**H**" command; the "**<**", "**>**", or "**=**" response to the "**%**" command; the "**<**", or "**>**" output when the VP9000 detects Encoder/Motor count mismatch; the "**W**" when a "**U6**" command is executed.

This command would be beneficial when the VP9000 is used with a Serial to IEEE488 bus converter. The terminator on the Converter should be set to "CR".

NOTE: A <cr> will not be added to the "**B**" or "**J**" response to a "**V**" command.

The "**G**" command is also used as **G**o when in Hold mode ("**H**" or "**U6**" command). **G**o after waiting or holding. The single character "G" is used to continue when the VP9000 is in a single step mode ("**H**" command) or when a "**U6**" command is being executed.

H Put the Controller on **H**old (single step mode). When the VP9000 receives the single character "**H**" the Hold Flag will be set. With the Hold Flag set, a "running" program stops after each command has completed it's operation, and a ":" is sent to the host. When stopped, the "**X**", "**Y**", "**Z**", and "**T**" commands can be used to read motor position. A "G" will cause the program to continue to the next operation. An "**H**" toggles the flag off and the program continues as normal. The "**K**" terminates the program and clears the Hold Flag. This Command allows single stepping through a program for debugging or as a program interrupt from the host.

<**Back Space**> (ASCII 8) Deletes the current partial value "keyed-in" when using the VP9000 with a terminal or computer running a terminal program.
NOTE: Any ASCII character greater than the value 57 will function as a delete.

X Send position of motor 1 to the host. When the VP9000 receives the single character "**X**" it will transmit the value from it's motor 1 Absolute Position Register. Below is what the host would receive if motor 1 is at negative 1200. This command can be used when the motor is indexing, however speed should be limited to 800 steps/sec. and the baud rate should be 9600. See the "**N**" command for information on zeroing the Absolute Position Registers.

-0001200<cr>

Y Send position of motor 2 to the host. When the VP9000 receives the single character "**Y**" it will transmit the value from it's motor 2 Absolute Position Register. Below is what the host would receive if motor 2 is at positive 9201. This command can be used when the motor is indexing, however speed should be limited to 800 steps/sec. and the baud rate should be 9600. See the "**N**" command for information on zeroing the Absolute Position Registers.

+0009201<cr>

Z Send position of motor 3 to the host. When the VP9000 receives the single character "**Z**" it will transmit the value from it's motor 3 Absolute Position Register. Below is what the host would receive if motor 3 is at negative 20. This command can be used when the motor is indexing, however speed should be limited to 800 steps/sec. and the baud rate should be 9600. See the "**N**" command for information on zeroing the Absolute Position Registers.

-0000020<cr>

T Send position of motor 4 to the host. When the VP9000 receives the single character "**T**" it will transmit the value from it's motor 4 Absolute Position Register. Below is what the host would receive if motor 4 is at negative 200000. This command can be used when the motor is indexing, however speed should be limited to 800 steps/sec. and the baud rate should be 9600. See the "**N**" command for information on zeroing the Absolute Position Registers.

-0200000<cr>

M Request **M**emory available for the currently selected program. The VP9000 will send the number of bytes that are unused of the current program. The value will be 000 to 256 followed by <cr> (carriage return).

Request the **number** of the currently selected motor. The VP9000 will send the number of last motor selected. The value will be 1 to 4 followed by <cr> (carriage return).

% Requests encoder / motor status from last index, When using the Encoder Interface Module with a 100 count encoders on the motor(s), the VP9000 does a count and compare between encoder pulses and motor counts (motor counts = 1/4 of steps). The maximum index distance the VP9000 can do a comparison is 524,284 steps. **NOTE:** The VP9000 will assume a stall has occurred if a single index is greater than 524,284 steps.
In response to this command the VP9000 sends one of the following characters:

- **=** when motor counts and encoder counts are **equal**
- **<** when encoder counts are **less** than motor counts (motor stall)
- **>** when encoder counts are **greater** than motor counts (motor overshoot). Motor overshoot can occur when a motor is interrupted when indexing at high speed (over 1000 steps / sec.) The interruption can be from encountering a limit switch; the host sending a "**K**", "**X**", "**Y**", "**Z**", or "**T**" command.

The proper time to use the "**%**" command would be after the host receives the "^" prompt, after each index, when controlling the VP9000 in an interactive mode. The next step would have the host either stopping and warning the user of a stall condition, and (or) homing and zeroing the motor to it's limit switch.

CAUTION: If the motor stalls, the cause should be remedied, here is a list of possible conditions that can cause stalling:

a) Motor speed too high
b) Acceleration too high
c) Excessive load
d) Over-tightened leadscrew nut adjustment
e) Wide temperature fluctuations (environmental)
f) Leadscrew lacking lubricant
g) Worn contacts or terminals have spread on motor cable connector from frequent connecting / disconnecting
h) Broken motor cable wire(s)
i) Slider (carriage) lacking lubricant
j) Slider overly-tight, due to over-tightened mounting screws or debris in dovetail ways
k) Unbalanced or shifting load or unpredictable external loads / forces
l) Damaged motor drive circuitry or bad motor (rare)

The other method to detect a motor stalled condition is to set stop-on-encoder-mismatch ON (refer to "Appendix A" (p.53) for setting from the Setup Submenu) and looking for a "<" or ">" character coming from the VP9000. The host can send a "**#**" command to determine which motor stalled or overshot.

The following two commands are only used for VP9000 controllers that are daisy-chained together.

{*x*} Send commands *x* to the next VP9000 in the "chain", *x* are any of the previous commands. Refer to the section "Daisy-chaining VP9000 Controllers" (p.23) for more information.

& Enable multiple VP9000s that are daisy-chained. The "**&**" command is used in place of the "**E**", "**F**", or "**G**" command when VP9000 controllers are daisy-chained together. Refer to the section "Daisy-chaining VP9000 Controllers" (p.23) for more information.

Troubleshooting Procedure

Refer to "Appendix B" (p.56) for information on troubleshooting error messages that are displayed on the front panel.

SYMPTOM	POSSIBLE CAUSE	CORRECTIVE ACTION
Power light does not "light" when VP9000 is switched on.	Line cord not fully plugged-in.	Check line cord at power receptacle on back of Controller for a tight connection.
	Bad fuse	Check fuse in power receptacle.
Motor does not operate.	Limit switch(es) circuit **open** or switches improperly connected or missing.	Check limit switches for proper action and connection, or if no switches are present, disable the limits from the Setup Menu.
On-Line light flashes continuously (on for 0.1 sec, off for 0.1 sec.) after power-up & nothing is on display.	Loose display cable or faulty display.	Check cable or replace display if necessary.
On-Line light flashes continuously (on for 0.1 sec, off for 1.0 sec.) after power-up.	Faulty U4 or U1 device	Replace device U4, U1
On-Line light flashes continuously (on / off / on for 0.1 sec, off for 1.0 sec.) after power-up.	Faulty U5 or U1 device	Replace device U5, U1

Controller does not come On-Line when sent "E","F", or "G".	RS-232 may not be connected correctly.	Trace Transmitted Data, Received Data, and Signal Ground wires from your computer to the VP9000.
	Your computer or terminal is not sending upper case letters.	Transmit only upper case letters. The Controller will not respond to lower case.
	A Mouse or FAX Modem Driver is controlling the same COM port as the VP9000 is connected.	Change to a different COM port, or remove the offending driver from the PC's AUTOEXEC.BAT or CONFIG.SYS file.
	Your computer may require a high on its Data Set Ready (DSR) line.	Check with the computer manufacturer to see if the DSR line must be pulled high, or if it can be disabled in software.
VP9000 does not come On-Line or program does not operate.	The RS-232 parameters are not set properly, or a device in your Computer is set to the same IRQ as your COM port.	Match the RS-232 settings on the VP9000 to those of your computer or terminal.
Your computer does not receive data from the Controller.	Your computer may require a "high" on its Request To Send (RTS) line.	Check with the computer manufacturer to see if the RTS line must be pulled high, or if it can be disabled in software.
Motor stalls, it does not move at all.	Inertia in system is too high, or mechanism has seized.	Hand rotate the system to locate any binding. A larger motor, or a different ratio (pitch) may be required.
Motor stalls, after rotating slightly.	Acceleration too high.	Use a lower acceleration /deceleration.
	Motor cannot overcome friction or load.	Check mechanism for ease of movement. Load will have to be reduced or counterbalanced.
Motor stalls before reaching maximum speed.	Motor torque decreases as it's speed increases.	Reduce steps/sec. setting of motor.

With Motor removed from equipment, Motor runs erratically at lowest speeds, at mid speed it has no torque (stalls).	Power to motor too high for a no load condition	Use the "**S**" speed command (70% power) instead of the "**SA**" speed command.
	Broken motor cable.	Check cable and connector for broken wires and repair breaks.
	Terminals on motor cable connector have poor contact due to frequent unplugging	Replace socket terminals on motor cable connector.
	Faulty TIP142 motor drive transistor(s).	Return to Velmex, Inc. for Service.
	Faulty Motor (Rare).	Try another motor.
Motor or system resonates (vibrates loudly).	The motor speed is the motor or system's natural resonant frequency (common at low speeds).	Increase or decrease speed (use "**S**" command) to avoid resonance points. A damper or flywheel on the motor shaft or lead screw may dampen the resonance.
Controller stops operating for no apparent reason.	Inductive surge on AC power-line.	Isolate or remove any equipment that may be putting "spikes" on the power-line.
	Inductive / Static surges coming in the I/O connections such as User input 4.	Make sure all external equipment connected is properly grounded and inductive loads are isolated from the VP9000.
The VP9000 Display shows black boxes or partial information in the wrong position.	A high static discharge occurred to the Controller.	Reset (Power off / on) and isolate the static electricity, if it occurs frequent, shield and ground all I/O cables.
When running in an interactive mode with a host, some move commands do not get executed.	The host is not waiting for the "^" after running an index.	Always program the host to look for the "^" after every run.
	Limit switch restricting motion.	Check limit switches for proper actuation.
Jog / slewing of motors not functional.	VP9000 is not in Jog / slew mode.	VP9000 must be off-line (On-Line off) and must not be in a menu.

Specifications

Functional

Packaged Controller / Driver, using microcomputer control of stepping motors. Operates one to four (dependent on model) motors, one-at-a-time. Bi-level motor drives with settable holding torque for motor one.
Interactive limit switch inputs (TTL), (CW and CCW for each axis).
User input 1 active high (0V to +3V min., -25V to +25V max.), Input 2 is TTL active low, and two user outputs (0 or +5V, 10 mA sinking and 3 mA sourcing capability).
Wide viewing angle, 2 line x 24 character, backlit, super-twist LCD readout for motor position display and data selection. Six key calculator quality keyboard for cursor control, display selection, and 0 entry.
Front panel programmable or through full-duplex RS-232-C; 300,1200,4800,9600 Baud (settable), 7 Data bits, Even parity, 2 Stop bits, ASCII; special configurations with 8 data bits, odd or no parity available on request.
User available NVRAM for program storage is 7936 bytes.
Remote Run, Reset, Two Interrupt inputs, Inputs to count quadrature converted pulses from multiplexed encoders, Remote pushbutton controller with coil cord.
Ten-foot motor and limit switch cables with connectors.

Motor Compatibility

American Precision 23D-6108A,23D-6209A,23D-6309A,34D-9109A,34D-9209A,34D-9311A
Bodine Electric 2430,2530,2431,2531,2411,2511,2433,2533,2434,2534,2435,2535
Superior Electric M061-LS08,M062-LS09,M063-LS09,M091-FD09,M092-FD09,M093-FD11
Vexta PX245-01,PK245-01,PX245-02,PX245M-01,PK245M-01,PX245M-02

Consult Velmex to determine equivalent settings for motors not listed above.

Physical

Weight: 22 lbs. (10 kg)
Height: 5.2 inches (13.2 cm)
Width (without handles): 8.5 inches (21.6 cm)
Depth (without handles): 14.3 inches (36.3 cm)

Electrical Requirements

90 to 130 VAC 50/60Hz, 190 watts

Environmental

35º to 95º F (2º to 35º C) Convection cooled

Models

Cat. # 4-911, Model # VP9001 One motor version
Cat. # 4-912, Model # VP9002 Two motor version
Cat. # 4-913, Model # VP9003 Three motor version
Cat. # 4-914, Model # VP9004 Four motor version

Options

Cat. # 4-918, 19" Rack Mount Extension Plates
Cat. # 4-920, Encoder Interface Module (Interfaces up to 4 encoders to the VP9000)

Appendix A - Setup Submenu

The Setup Submenu is selected by pressing the "SEL" key when the cursor is over the "S" of "Setup" on the Main Menu (the Main Menu is selected by pressing the "SEL" key when the VP9000 is off-line). The Setup Submenu looks like this:

```
Mtrsel RS-232 Bcomp Encd
Trq1 Lmts Scale Defaults
```

Mtrsel (Motor Select) Submenu (selected by pressing the "SEL" key when the cursor is over the "M" of "Mtrsel" on the Setup Submenu; the VP9000 will automatically enter this menu when no motors have been selected)

```
Mtr 1234: Not Selected
(No Motor)
```

Purpose: To select the motor attached to each axis of the VP9000. The motor selected "tells" the VP9000 how long to apply electrical energy to the motor. The current applied to the motor will therefore be a function of motor inductance. Since all motors have different inductance, it is imperative that the exact Manufacturer and Model be selected.

CAUTION: Select the Motor currently attached for each axis.
Always reselect when a new or different motor is "plugged" into an axis. Improper selection will result in Controller, Motor overheating, and Severe Damage, not covered by the Warranty !

Keys used to select axis (moves cursor over motor number): →,←
Key used to select motor Model: "SEL"
Key used to select motor Manufacturer: ↓
Key used to select "No Motor": "0"
Key to exit Submenu (current setting displayed stored in NVRAM after confirmation): ↑

Locate the Manufacturer and Model number from each motor attached to the VP9000. The Manufactures are:

AMERICAN PRECISION industries, inc. CONTROLS DIVISION	(AMERICAN PREC.)
BODINE ELECTRIC COMPANY	(BODINE ELEC.)
SUPERIOR ELECTRIC	(SUPERIOR ELEC.)
VEXTA oriental motor co.	(VEXTA)

Now locate the Model number.
Below are examples of where the Model number is located for the different Manufactures.

AMERICAN PREC: **MODEL**

34D-9109A

BODINE ELEC: **NO** 2431

SUPERIOR ELEC: **TYPE**

M091-FD09

VEXTA: **MODEL**

PX245-01

Write down the motors here for reference:

Axis **Manufacturer** **Model**

Mtr 1 _____

Mtr 2 _____

Mtr 3 _____

Mtr 4 _____

The ↓ key advances the **Manufacturer**, and the "**SEL**" key advances the **Model**.

After selecting motor(s) press the ↑ key. The next menu to appear will be the confirmation menu. Double check to make sure the motor selected is the same as the one attached to the VP9000. Use the ← key to put the cursor over "Yes", and press the "SEL" key. The VP9000 will request confirmation for each new motor selection.

RS-232 Submenu (selected by pressing the "SEL" key when the cursor is over the "R" of "RS-232" on the Setup Submenu)

```
7 Data Bits, Even Parity
2 Stop Bits, Baud = 9600
```

Purpose: Change RS-232 baud rate, display current baud rate, and show permanent RS-232 parameters.
Keys used to select new value: →,←, "SEL"
Keys to exit Submenu (current setting displayed stored in NVRAM): ↑,↓, "0"

Bcomp (Backlash Compensation) Submenu (selected by pressing the "SEL" key when the cursor is over the "B" of "Bcomp" on the Setup Submenu)

```
      Backlash
   Compensation is OFF
```

Purpose: Change the state of backlash compensation, and show current setting. When backlash compensation is ON, and a motor makes a negative Index, 20 steps will be added to the Index. The Motor will then immediately reverse, indexing positive 20 steps.
Keys used to toggle ON/OFF: →,←, "SEL"
Keys to exit Submenu (current setting displayed stored in NVRAM): ↑,↓, "0"

Encd (Encoder) Submenu (selected by pressing the "SEL" key when the cursor is over the "E" of "Encd" on the Setup Submenu)

```
Stop on Encoder Mismatch
Motor:   1 2 3 4  is OFF
```

Purpose: Enable/disable automatic stop on Encoder/Motor count mismatch, and show current settings. The VP9000 will send a ">" or "<" character to the host at the end of an index, instead of the "^", when encoder and motor counts are not the same. Refer to the "**%**" command on page 46 for more information.

NOTE: Stop on Encoder Mismatch should only be ON if there is an encoder on a motor and the encoder is properly connected to the optional Encoder Interface Module.

Keys used to select motor: →,←
Key used to toggle ON/OFF: "SEL"
Keys to exit Submenu (current setting displayed stored in NVRAM): ↑,↓, "0"

Trq1 (Motor 1 Holding Torque) Submenu (selected by pressing the "SEL" key when the cursor is over the "T" of "Trq1" on the Setup Submenu)

```
Motor 1 Holding Torque
            is FULL OFF
```

Purpose: Change the state/show current setting of holding torque on Motor 1. When holding torque is HALF ON, and Motor 1 is at a standstill, (with VP9000 "on" also) Motor 1 will be electrically energized to the nearest half-step position (every 1.8°). When Motor 1 Holding Torque is FULL ON, Motor 1 will stay fully energized at the current step position (every 0.9°). When Motor 1 Holding Torque is FULL OFF, Motor 1 will be unenergized 130 milliseconds after indexing and power-up-lock-in.

NOTE: Use the "ON" settings only if necessary, the HALF ON consumes less energy than the FULL ON setting. To conserve energy and reduce heating set Holding Torque to FULL OFF.

NOTE: Holding torque should only be ON if needed to prevent a load from "back-driving", which is common on a 2.5 or 1 pitch (sometimes on a 5 pitch) lead screw supporting a load vertically.
DANGER: When the VP9000 is "turned-off" the load may drop causing mechanical damage and/or personal injury.

Keys used to toggle HALF ON/FULL ON/FULL OFF: →,←, "SEL"
Keys to exit Submenu (current setting displayed stored in NVRAM): ↑,↓, "0"

Lmts (Limit Switches) Submenu (selected by pressing the "SEL" key when the cursor is over the "L" of "Lmts" on the Setup Submenu)

```
   Limit Switch Inputs
Motor:   1 2 3 4 are ON
```

Purpose: Enable/disable limit switch inputs, and show current settings. Limit Switch Inputs should be OFF if there are not any limit switches for the motor.

Keys used to select motor: →,←
Key used to toggle ON/OFF: "SEL"
Keys to exit Submenu (current setting displayed stored in NVRAM): ↑,↓, "0"

Scale Submenu (selected by pressing the "SEL" key when the cursor is over the "S" of "Scale" on the Setup Submenu)

```
Display Motor : 1 2 3 4
Scaled for Steps (0.9˚)
```

Purpose: Select a lead screw or rotary table ratio to display position in "real" units (inches, millimeters, degrees, instead of motor steps), and show current settings. Refer to the table in the "Reference" section (p.25) for more information on the leadscrew designations.

Keys used to select motor: →,←

Key used to select: "SEL"

Keys to exit Submenu (current setting displayed stored in NVRAM): ↑,↓, "0"

<u>Defaults (Total Reset) Submenu</u> (selected by pressing the "SEL" key when the cursor is over the "D" of "Defaults" on the Setup Submenu)

```
Reset Everything To
Factory Defaults ?   Yes
```

Purpose: Totally reset everything to factory defaults and clear all programs from memory. If the "SEL" key is pressed, the following menu will appear, giving the user one more chance:

```
ALL Programs Will Be
ERASED ! Continue ?  Yes
```

Key used to Reset: "SEL"

Keys to exit Submenu (no change to programs or settings stored): ←,→,↑,↓, "0"

If "SEL" is pressed, this next menu will appear:

```
NVRAM ERASED !    Press
     Any Key to Continue
```

Press any key and the VP9000 will then do the total reset.

The following are the default settings after a total reset:

1. All programs cleared from memory.
2. Motor speeds will be 2000, Accelerations will be 2.
3. All motor selections will be set to "No Motor".
4. All setup settings will be set as shown in the previous submenus.
5. Main Menu lockout off.

Appendix B - Error Messages on Front Panel Display

```
MOTOR SELECT ERROR !
 RESET Circuit Breaker
```

Cause: Over-current on motor has tripped the circuit breaker.
DANGER: Improper motor selection in the Setup Submenu causes this condition. The VP9000 will set all axes to "No Motor" selected when this fault occurs. Locate the circuit breaker on the rear of the of the VP9000, it is marked "RE-CIRK-IT". Reset it by pushing the white indicator in flush with the end of the circuit breaker. Refer to "Appendix A" (p.51, Motor Select Submenu) for proper selection of motors.

```
PROGRAM NUMBER OUT OF
RANGE, MUST BE 0 TO 30 !
```

Cause: User has tried to select a program number that does not exist. The VP9000 will go off-line when this error occurs.

```
RS-232 PARITY ERROR !
CHECK RS-232 SETTINGS !
```

Cause: Parity, Baud rate, Data bits, or Stop bits setting(s) is not the same between host and VP9000. **NOTE:** On-Line light flashes, and to recover from this error the Controller must be turned-off or reset.

```
RS-232 FRAMING ERROR !
DID THE HOST POWER DOWN!
```

Cause: BREAK Signal (happens when host is turned-off) detected on RS-232 communication line; or Parity, Baud rate, Data bits, or Stop bits setting(s) not the same between host and VP9000. **NOTE:** On-Line light flashes, and to recover from this error the Controller must be turned-off or reset.

```
OPEN RUN INPUT TO
CONTINUE !
```

Cause: An input is detected on the Run input (I/O,4). The VP9000 checks for the Run input every time it is powered-up. The purpose of this check is to prevent unexpected starts when the VP9000 is powered-up. By "opening" the Run input circuit, the VP9000 will continue as normal.

```
OPEN JOG INPUT TO
CONTINUE !
```

Cause: An input is detected on the Jog controller. The VP9000 checks for the Jog inputs every time it is powered-up. The purpose of this check is to prevent unexpected jog/slewing when the VP9000 is powered-up. By releasing the Jog button(s), the VP9000 will continue as normal.

```
OPEN INPUT 3 TO
CONTINUE !
```

Cause: A low is detected on User input 3 (I/O,7). The VP9000 checks for a low on input 3 every time it is powered-up. The purpose of this check is to prevent unexpected lock-ups in a program due to waiting for input 3 to go high. By "opening" the input 3 circuit (I/O,7), the VP9000 will continue as normal.

```
NVRAM ERASED !    Press
     Any Key to Continue
```

Followed by

```
BAD NVRAM
```

Cause: The NVRAM has been damaged or is defective (device U3).

NOTE: The first message will always be displayed when a total reset is done from the Setup Submenu. The first message may also occur when the AC Line voltage is below 90V when the Controller is powered-up, the NVRAM goes into protect mode under such low voltage (logic voltage below 4.8V) conditions.

```
BAD RAM
```

Cause: The on-chip RAM of device U1 has been damaged or is defective. This error would occur at time of power-up. Every time the VP9000 is powered-up, the RAM is automatically tested.

```
OUT OF PROGRAM MEMORY
FOR THIS PROGRAM !
```

Cause: Too many commands in the current program (exceeded the 256 byte limit per program). The VP9000 will go off-line when this error occurs.

```
OUT OF JUMP-&-COME-BACK
MEMORY, TOO MANY "JMxs"!
```

Cause: More than 30 "**JM*x***" commands active at one time. The VP9000 will go off-line when this error occurs. Using the "**JM*x***" command like a "**J*x***" command will produce this error. For example, the following program is an infinite loop that will cause memory overflow:

PM-0,JM1,PM-1,JM0,R

```
OUT OF LOOP COUNT MEMORY
TOO MANY NESTED LOOPS !
```

Cause: More than 33 loop commands active at one time. The VP9000 will go off-line when this error occurs. This error will occur if there are greater than 33 consecutive loop commands encountered per "run".

```
A STALL HAS BEEN
DETECTED ON MOTOR x
```

x is the motor # (1,2,3,4)

Cause: The VP9000 has detected encoder counts to be less than Motor x counts. Motor x must have been enabled (see Setup Submenu, "Appendix A," p.51) to stop and display this error message. This error is due to either an encoder not attached or the motor has stalled. Refer to the "**%**" command on page 46 for more information on motor stalling.

```
OVERSHOOT OR BAD ENCODER
DETECTED ON MOTOR x
```

x is the motor # (1,2,3,4)

Cause: The VP9000 has detected encoder counts to be greater than Motor x counts. Motor x must have been enabled (see Setup Submenu, "Appendix A," p.51) to stop and display this error message. This error is due to either an encoder error/failure or the VP9000 being interrupted by the host while indexing. Refer to the "**%**" command on page 46 for more information on motor overshoot.

```
CONTINUOUS INDEX ERROR
MOTOR IS NOT THE SAME !
```

Cause: The VP9000 has detected more than one motor selected between Continuous Indexing commands. The VP9000 will go off-line when this error occurs. Refer to the "**U7**" command on page 35 for more information on Continuous Indexing.

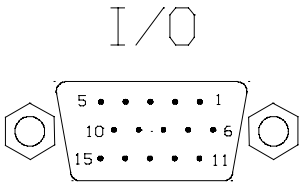
```
CONTINUOUS INDEX ERROR
U9 COMMAND MISSING !
```

Cause: The VP9000 has detected the "**U9**" command missing while starting Continuous Indexing. The VP9000 will go off-line when this error occurs. Refer to the "**U9**" command on page 36 for more information on the proper ending of Continuous Indexing.

```
CONTINUOUS INDEX ERROR
COMMAND NOT ALLOWED !
```

Cause: More than six Speeds, Speeds with Loops, Acceleration, Pause, -Loop, or Jump command(s) are found in Continuous Indexes. The VP9000 will go off-line when this error occurs.

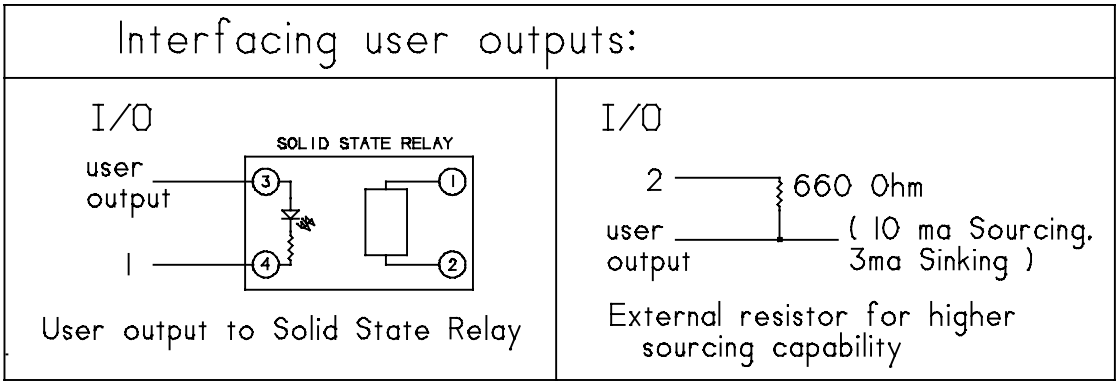
Appendix C - I/O Connections

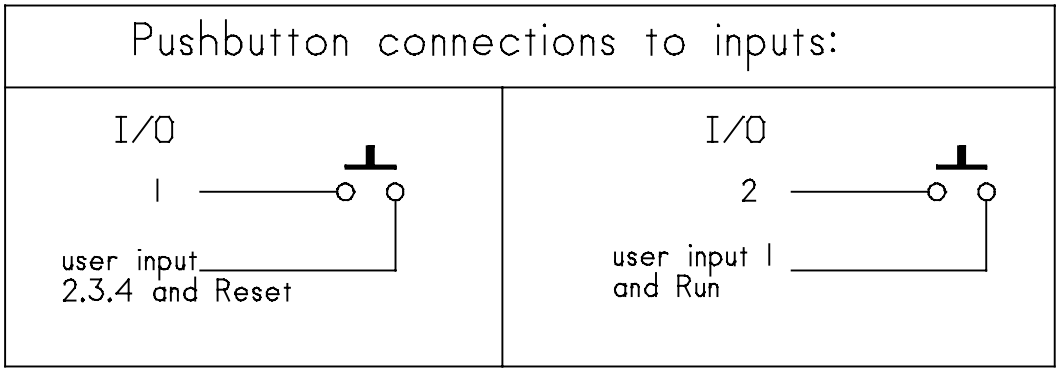


Pin #	Description	Function
1	Signal Ground (Gnd)	Common reference connection for all inputs and outputs
2	+5 Volt (+5V)	User available power (200 mA max.) for circuits and for powering Encoder Interface Module.
3	Signal Ground (Gnd)	Same as pin 1.
4	Run	When connected to pin 2 (+5V) for at least 1 ms, the current program will be executed, and is the input for the "**U90**" wait command.
5	User input 1	When connected to pin 2 (+5V) for at least 1 ms, the wait commands for user input 1 will be satisfied.
6	User input 2	When connected to pin 1 (Gnd) for at least 1 ms, the wait commands for user input 2 will be satisfied.
7	User input 3	Interrupts a wait on input 1,2, or "**U90**", and jumps to program 29, when a 20 ms low and 20 ms high occurs (debounced input suitable for a push-button switch connected to Gnd). Outputs 1 and 2 will be reset if they where held high by the wait.
8	User input 4	Interrupts an Index (by decelerating to a stop, motor position saved at time of interrupt, saved position can be requested later with the "*****" command) or a wait on input 1,2, or "**U90**", and jumps to program 30, when connected to Gnd (high to low transition). Outputs 1 and 2 will be reset if they where held high by a wait.
9	Reset	When connected momentarily to pin 1, (Gnd) the VP9000 will reset to the power-up state. **CAUTION**: Only connect with a push-button switch, or "dry" relay contacts.
10	Encoder Up Cnt	Input from optional Encoder Interface Module.

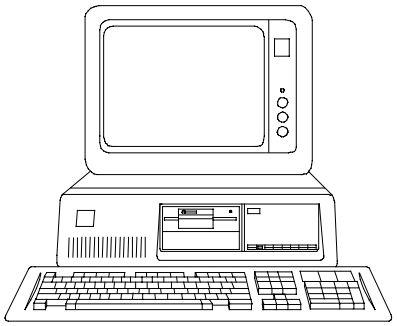
11	Encoder Dwn Cnt	Input from optional Encoder Interface Module.
12	User Output 1	User programmable output, either 0V (default on power-up) or +5V. This output can drive 4 TTL loads directly (3mA sourcing, 10mA sinking).
13	User Output 2	User programmable output, either 0V (default on power-up) or +5V. This output can drive 4 TTL loads directly (3mA sourcing, 10mA sinking).
14	Motor Sel A	Output to select encoder from optional Encoder Interface Module.
15	Motor Sel B	Output to select encoder from optional Encoder Interface Module.

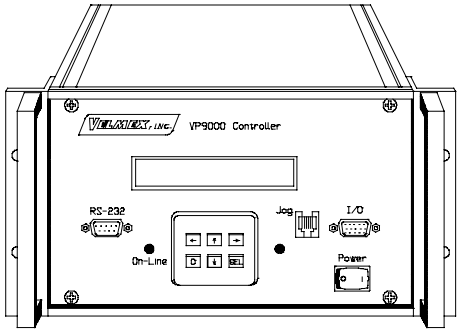
IMPORTANT: All inputs and outputs are TTL levels (0 to +5V) except for Run and Input 1. The Run input and Input 1 can be interfaced to voltages as high as +25V.

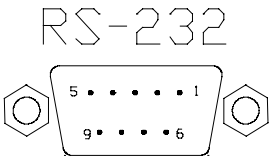




Appendix D - RS-232 Cable Configuration







IBM PC,XT,AT, & Compatibles
(COM1,COM2,COM3,COM4)

VP9000 (RS-232)
7 Data, Even Parity, 2 Stop

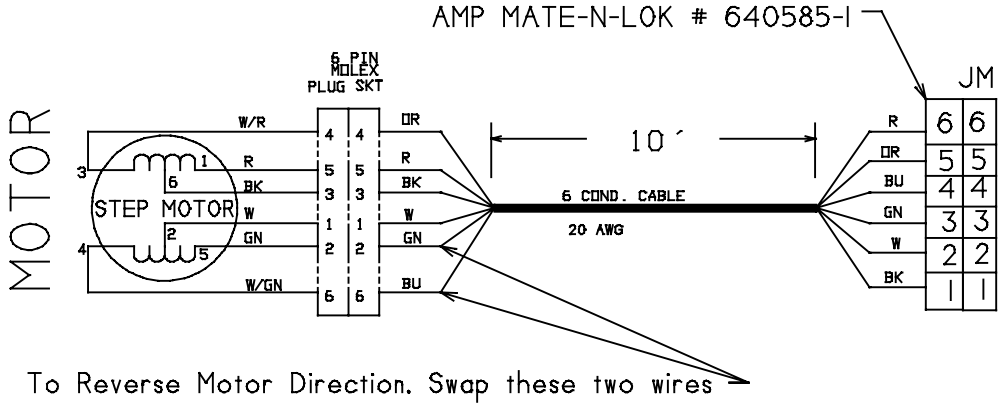
Serial (9 Pin)			RS-232
RX	**2**	———	**2** TX
TX	**3**	———	**3** RX
GND	**5**	———	**5** GND
DTR	**4**	———	**4** ┐
DSR	**6**	———	**6** ┘
RTS	**7**	———	**7** ┐
CTS	**8**	———	**8** ┘

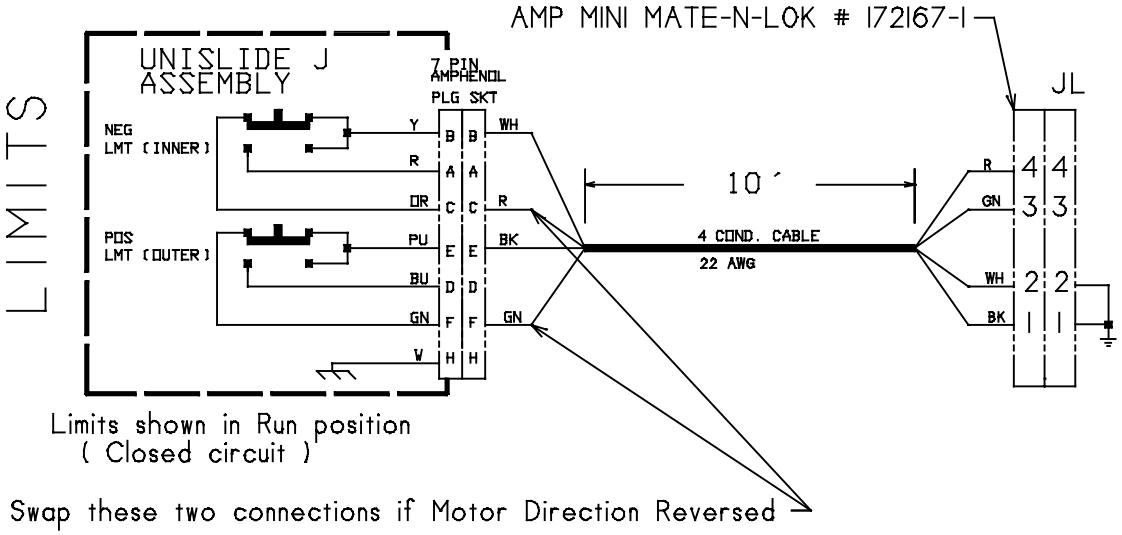
Serial (25 Pin)			RS-232
RX	**3**	———	**2** TX
TX	**2**	———	**3** RX
GND	**7**	———	**5** GND
DTR	**20**	———	**4** ┐
DSR	**6**	———	**6** ┘
RTS	**4**	———	**7** ┐
CTS	**5**	———	**8** ┘

NOTE: Make certain your program or terminal is set at 7 data, 2 stop bits and even parity, and the same baud rate (baud rate is factory set to 9600) as the VP9000 (see "Appendix A" (p.53) to change baud rate on the VP9000).

RS-232 Loop-Back Self Test (sends a "U" out TX, waits for echo on RX, 25000 times)
The TX and RX on the VP9000 can be tested by putting a jumper from pin 2 to 3 when the VP9000 is off. Hold the ↑ key on the VP9000 down while turning the VP9000 on. Hold the ↑ key until the On-Line light comes on. If the jumper is connected correctly, and the RS-232 port is functioning properly the On-Line light will go out in 28 seconds (assuming baud is set to 9600).

Appendix E - Motor and Limit Switch Cable Configuration

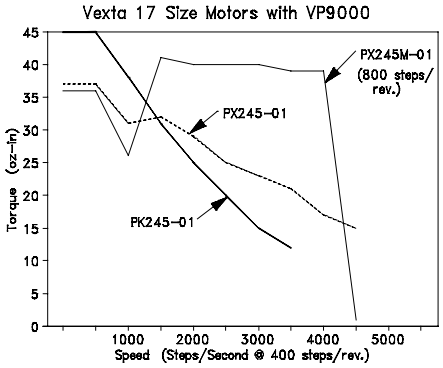


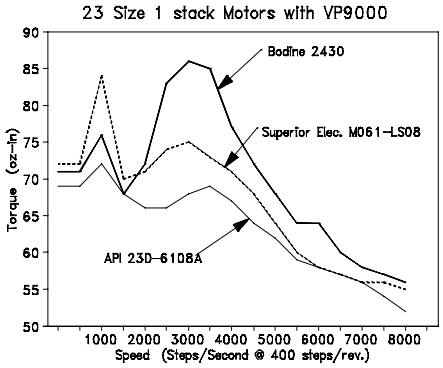


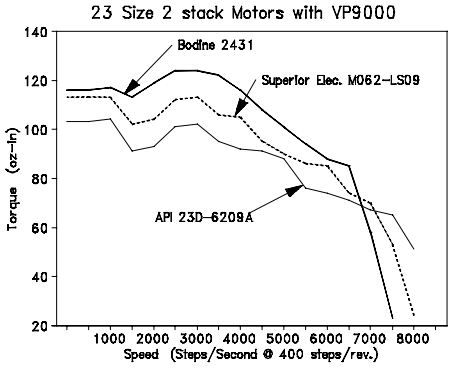
CAUTION: Motor cables should **never** be bundled together with the Limit Switch, or any I/O cabling. **Never** put any of the VP9000's cables with power cables in a common electrical conduit or ducting. **Always** keep Limit Switch and I/O cables at least 3 inches from Motor and Power cables.

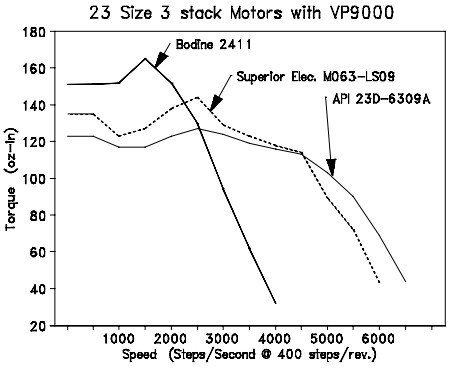
CAUTION: Maximum Allowable Motor Cable length is 50 feet (15.2 meters) with 20 AWG wire.

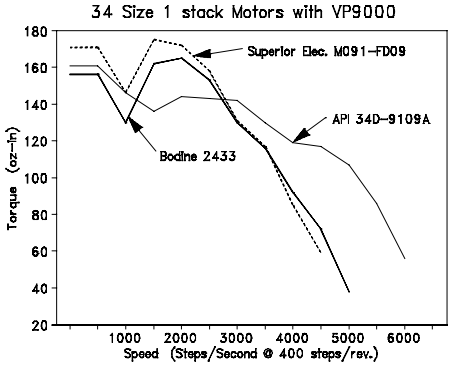
Appendix F - Motor Performance (100% Power)

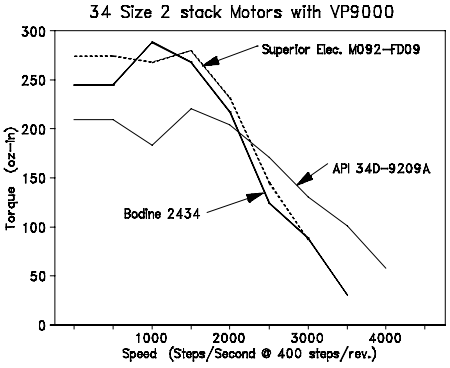


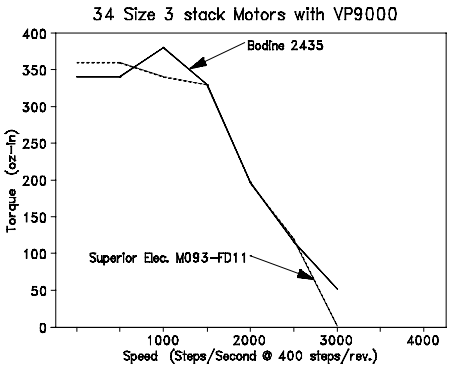












Appendix G - Main Menu Lockout Toggle

Normally the user has complete access to the Main Menu and all the submenus from the VP9000's keyboard. However, to run the VP9000, after initial programming, it is not necessary (and may not be desirable) for the operator to have access to any submenu other than the Program Submenu.

The Main Menu lockout feature can be set to deny operator access to all menus other than the Program Submenu.

When the Main Menu lockout feature is on, the user/operator can only access the Program Submenu, and programs can not be deleted from the Program Submenu.

To toggle the Main Menu lockout "on" or "off":

1. Switch Power off.
2. Press and simultaneously hold down the ← and → keys on the VP9000.
3. Switch Power on.
4. Hold the ← and → keys down until the Main Menu (lockout off), or the Program Submenu (lockout on) is displayed.

The VP9000 keeps the lockout setting in it's non volatile memory (NVRAM) to maintain the last setting every time the VP9000 is powered-up. The only way to change the lockout setting is with the above procedure.

The lockout setting has no effect on programming over the RS-232, all RS-232 commands are fully functional when lockout is on.

Limited Two Year Warranty

Velmex warrants this Controller against defects in material and workmanship for a period of two years from date of shipment. In case of defect, Velmex will repair this Controller without charge. The user will be responsible for shipment to Velmex, and will be charged a minimum of $45.00 for non-warranty service.

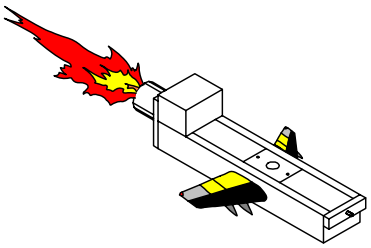
For service under this warranty:

1. Contact Velmex to obtain a Return Authorization number (Tel. 716-657-6151).
2. Pack Controller in the original container or equal.
3. Enclose a description of the problem, along with the return address, the name of the end user, his/her telephone and fax number.
4. Ship Controller Prepaid and insured to :

   Velmex, Inc
   7550 State Route 5&20
   Bloomfield, NY 14469

Exclusions and Limitations:

This warranty covers step motor controllers manufactured by Velmex, Inc. External electrical cables and connectors are not covered for wear and breakage. This warranty does not extend to any damage or malfunction resulting from misuse, neglect or accident. Except for any implied warranty, this warranty contains the entire obligation of Velmex, and the remedies described above are the exclusive remedies under this warranty or any implied warranty. The duration of any implied warranty is limited to two years. **In No Event Shall Velmex Be Liable For Incidental or Consequential Damages.**



VP9000

Quick Start Guide

1. Connect cables (p.4)

2. Turn-on the VP9000

3. Refer to "Appendix A" (p.51) to select motors attached to VP9000; if limit switches are not available, refer to p.51 & p.54 to disable limit inputs

4. Run the VP9000:

    a. Jog motors with remote pendant (p.5)

    b. Control with host computer (p.13 and p.20), view the file "VPREADME.DOC" on the "VP9000" diskette for a complete list of example programs and utilities

RS-232 Interactive Mode

9600 Baud, 7 Data, Even Parity, 2 Stop

1. The host puts the VP9000 On-Line by sending an "**F**"
2. The host sends a "**N**" to zero position registers if necessary
3. The host sends speed, and acceleration if necessary
4. The host sends an Index ("**I**" command, p.25 & p.26)
5. The host sends a "**R**" to start the Index
6. The host then will wait until it receives a ready prompt ("^") from the VP9000

NOTE: The VP9000 does not send a carriage return following the "^" unless the "**G**" command is used instead of "**F**" in step 1

7. The user's routine for outputting, measuring, etc. would be executed by the host
8. A "**C**" would be sent from the host to clear the previous Index command from the VP9000's memory
9. The process is repeated from step # 3

    c. Control / program at front panel (p.7)

Ver. 2.8 VP9000 Command Summary

I*m***M***x* Set steps to incremental **I**ndex motor CW (positive), *m*= motor# (1,2,3,4), *x*=1 to 16,777,215
I*m***M-***x* Set steps to incremental **I**ndex motor CCW (negative), *m*= motor# (1,2,3,4), *x*=1 to 16,777,215
IA*m***M**x Set **A**bsolute **I**ndex distance, m=motor# (1,2,3,4), x= ±1 to ±16,777,215 steps
IA*m***M0** **I**ndex motor to **A**bsolute zero position, *m*=motor# (1,2,3,4)
IA*m***M-0** Zero motor position for motor# *m*, *m*= 1,2,3,4
I*m***M0** **I**ndex motor until positive limit is encountered, *m*=motor# (1,2,3,4)
I*m***M-0** **I**ndex motor until negative limit is encountered, *m*=motor# (1,2,3,4)
S*m***M***x* Set **S**peed of motor (70% power), *m*= motor# (1,2,3,4), *x*=0 to 8000 steps/sec. 0 being 1 step/ 2 sec. **SA***m***M***x* is 100% power
A*m***M***x* **A**cceleration/deceleration, *m*= motor# (1,2,3,4), *x*=1 to 127; 1 is 1000 steps/sec^2, 2 is 2000 steps/sec^2, 127 is 127000 steps/sec^2, etc.
L0 **L**oop continually from the beginning or Loop-to-marker of the current program
LM0 Sets the **L**oop-to-marker at the current location in the program
LM-0 Resets the **L**oop-to-marker to the beginning of the current program
L*x* **L**oop from beginning or Loop-to-marker *x*-1 times (*x*=2 to 65,535), when the loop reaches its last count the non-loop command directly preceding will be ignored
L-*x* **L**oop from beginning or Loop-to-marker *x*-1 times, alternating direction of motor 1, when the loop reaches its last count the non-loop command directly preceding will be ignored
LA*x* **L**oop **A**lways from beginning or Loop-to-marker *x*-1 times (*x*=2 to 65,535)
LA-*x* **L**oop **A**lways from beginning or Loop-to-marker *x*-1 times, alternating direction of motor 1
LM-2 **L**oop once from beginning or Loop-to-marker reversing index direction of motor 2
LM-3 **L**oop once from beginning or Loop-to-marker reversing index direction of motor 1 and motor 2
P*x* **P**ause *x* tenths of a second, (*x*=0 to 65,535, 10 µsec pause when *x*=0) tenths of a millisecond when *x* is negative
PA*x* **P**ause *x* tenths of a second **A**ltering output 1 high for duration of the pause, tenths of a millisecond when *x* is negative
B*x* **B**acklash compensation, compensation on when *x*=1, off when *x*=0
O*x* Indicate limit switch **O**ver-travel to host, off when *x*=0, VP9000 sends "O" when *x*=1 and a limit switch is encountered
PM*x* Select **P**rogra**m** number *x*, *x*= 0 to 30
PM-*x* Select and clear all commands from **P**rogra**m** number *x*, *x*= 0 to 30
PM Request the number of the current **P**rogra**m**
J*x* **J**ump to the beginning of program number *x*, *x*= 0 to 30
JM*x* **J**ump to the beginning of program number *x* and come back for **M**ore after program *x* ends, *x*= 0 to 30
U0 Wait for a "high" on user input 1
U1 Wait for a high on user input 1, holding user output 1 high while waiting
U2 Enable Jog mode while waiting for an input
U3 Disable Jog mode while waiting for an input
U4 User output 1 "low" (reset state)
U5 User output 1 high
U6 Send "W" to host and wait for a "G" to continue
U7 Start of Continuous Index with pulse on output 2
U8 Start of Continuous Index sending "@" to the host
U9 End of Continuous Index
U10 Wait for a low on input 2
U11 Wait for a low on input 2, holding user output 2 high while waiting
U12 Wait for a keyboard key to jump to a program or continue: ← key to jump to program #21, → key to jump to program #22, ↓ key to jump to program #23, ↑ key to jump to program #24, 0 or SEL key to proceed in current program
U13 Wait for a button on Remote Control to jump to a program or continue: 1- key to jump to program #25, 1+ key to jump to program #26, 2- key to jump to program #27, 2+ key to jump to program #28, ▭ key to proceed in current program
U14 **U**ser output 2 low (reset state)
U15 **U**ser output 2 high
U16 Wait for ← key to be pressed on front panel keyboard
U17 Wait for → key to be pressed on front panel keyboard
U22 Wait for a keyboard key to jump to a program and come back, or continue: ← key to jump to program #21 and come back, → key to jump to program #22 and come back, ↓ key to jump to program #23 and come back, ↑ key to jump to program #24 and come back, 0 or SEL key to proceed in current program
U23 Wait for a button on Remote Control to jump to a program and come back, or continue: 1- key to jump to program #25 and come back, 1+ key to jump to program #26 and come back, 2- key to jump to program #27 and come back, 2+ key to jump to program #28 and come back, ▭ key to proceed in current program
U30 Wait for a high to low transition on user input 1
U31 Wait for a high to low transition on user input 1, holding user output 1 high while waiting
U32 Wait for 1- button to be pressed on Remote Control
U33 Wait for 1+ button to be pressed on Remote Control
U40 Wait for a low to high transition on input 2
U41 Wait for a low to high transition on input 2, holding user output 2 high while waiting
U50 Wait for a high and low on user input 1 with debouncing for a mechanical push-button switch
U51 Wait for a high and low on user input 1 with debouncing for a mechanical push-button switch, holding user output 1 high while waiting
U60 Wait for a low and high on user input 2 with debouncing for a mechanical push-button switch
U61 Wait for a low and high on user input 2 with debouncing for a mechanical push-button switch, holding user output 2 high while waiting
U70 Zero counts in user programmable counter
U71 Display user programmable counts "on"
U72 Display user programmable counts "off"
U73 Increment user programmable counter by one
U90 Wait for a high and low on the Run connection I/O,4 with debouncing for a mechanical push-button switch
Q **Q**uit On-Line mode (return to Jog/slew mode)
R **R**un currently selected program
N **N**ull (zero) motors 1,2,3,4 absolute position registers
K **K**ill operation in progress and reset user outputs
V **V**erify Controller's status, VP9000 sends "B" to host if busy, "R" if ready, or "J" if in the Jog/slew mode
C **C**lear all commands from currently selected program
D **D**ecelerate to a stop (interrupts current index in progress)
E **E**nable On-Line mode with echo "on"
F Enable On-Line mode with echo "o**FF**"
G Enable On-Line mode with echo off **G**rouping a <cr> with "**^**", "**<**", "**>**", "**=**", "**:**", "**W**", "**O**" responses; Also **G**o after waiting or holding
H Put Controller on **H**old (single step mode)
X Send position of motor 1 to host
Y Send position of motor 2 to host
Z Send position of motor 3 to host
T Send position of motor 4 to host
M Request **M**emory available for currently selected program
Request the **number** of the currently selected motor
% Requests encoder/motor status from last index, VP9000 sends: **=** when motor counts and encoder counts are **equal**, **<** when encoder counts are **less** than motor counts (motor stall), **>** when encoder counts are **greater** than motor counts (motor overshoot)
{*x***}** Send commands to the next VP9000 in the "chain", *x* are any of the above commands
& Enable multiple VP9000s that are daisy-chained
★ Request the position when the last motor that was interrupted by a "**D**" command or user input 4