USER'S MANUAL

INTELLIGENT MOTOR CONTROLLERS

SRX FAMILY

OREGON MICRO SYSTEMS, INC.
TWIN OAKS BUSINESS CENTER
1800 NW 169th PLACE, SUITE C100
BEAVERTON, OR 97006
PHONE 503-629-8081
FAX 503-629-0688
EMAIL sales@OMSmotion.com
WEB SITE http://www.OMSmotion.com

COPYRIGHT NOTICE

TRADEMARKS

DISCLAIMER

Part Number: E4MN-SRX-A
Revision A

TABLE OF CONTENTS

1. GENERAL DESCRIPTION

1.1. INTRODUCTION

The SRX family of intelligent motion controls can manage as many as 8 axes of coordinated or independent motion on one board conforming to the Eurocard specifications. They meet the RS-232 and RS-422 specifications for serial communication and are compatible with most PLC's through a parallel port. The SRX-4E can, for example, simultaneously control four axes of step motors while monitoring their actual position with the built in incremental encoder interface, while the SRX-8 manages 8 axes.

The SRX functions as a motion processor which can be connected to an RS-232 or RS-422 port on any computer, operated stand alone or as a PLC peripheral. It utilizes a 68000 microprocessor and patented proprietary technology to control direction of motion, acceleration, deceleration and velocity of an associated motor. In response to commands from the host computer, the SRX controller will calculate the optimum velocity profile to reach the desired destination in the minimum time while conforming to the programmed velocity and acceleration parameters. A block diagram of the SRX is shown in Section 9.

The SRX family of controllers use 'microstepping' techniques for increased position resolution and decreased low speed resonance. When combined with the appropriate driver and step motor, the SRX can divide the normal step angle into 250 discrete steps of 0.0072 degrees each or 50,000 steps per revolution. The Oregon Micro Systems drivers, models MD10A and MH10 are 10 microstep per step motor drivers which can be driven by the SRX family of controllers.

The SRX is easily programmed by sending commands as simple ASCII command strings over the serial port of virtually any computer. For a typical motion requirement of 1,000,000 pulses at 400,000 pulses/sec and an acceleration of 500,000 pulses/sec^2 the following string would be sent from the host computer to the SRX:

VL400000 AC500000 MR1000000 GO

For additional programming examples see Section 7.

1.2. FUNCTIONAL DESCRIPTION

The SRX, in response to commands from the host computer, provides controlled acceleration to a predefined peak speed followed by a constant velocity and controlled deceleration to a stop. This is achieved by calculating the optimum velocity 1024 times each second, providing a very smooth acceleration curve. This calculation is used to control a variable frequency pulse train which is derived from a crystal oscillator thus providing very accurate pulse rates.

The 68000 microprocessor calculates this velocity profile for each of the axes providing independent but synchronized (if desired) profiles for each axis. The SRX can perform a smooth coordinated move on up to eight axes using linear, parabolic or cosine velocity profiles. It can manage as many as eight independent or coordinated processes.

The SRX will calculate the optimum velocity profile to generate the desired move, while conforming to the acceleration and velocity data input by the host computer. This move will consist of a smooth acceleration, followed by a constant velocity section and a smooth deceleration to the desired position. A graph of a typical linear velocity profile is shown in Figure 1-1.

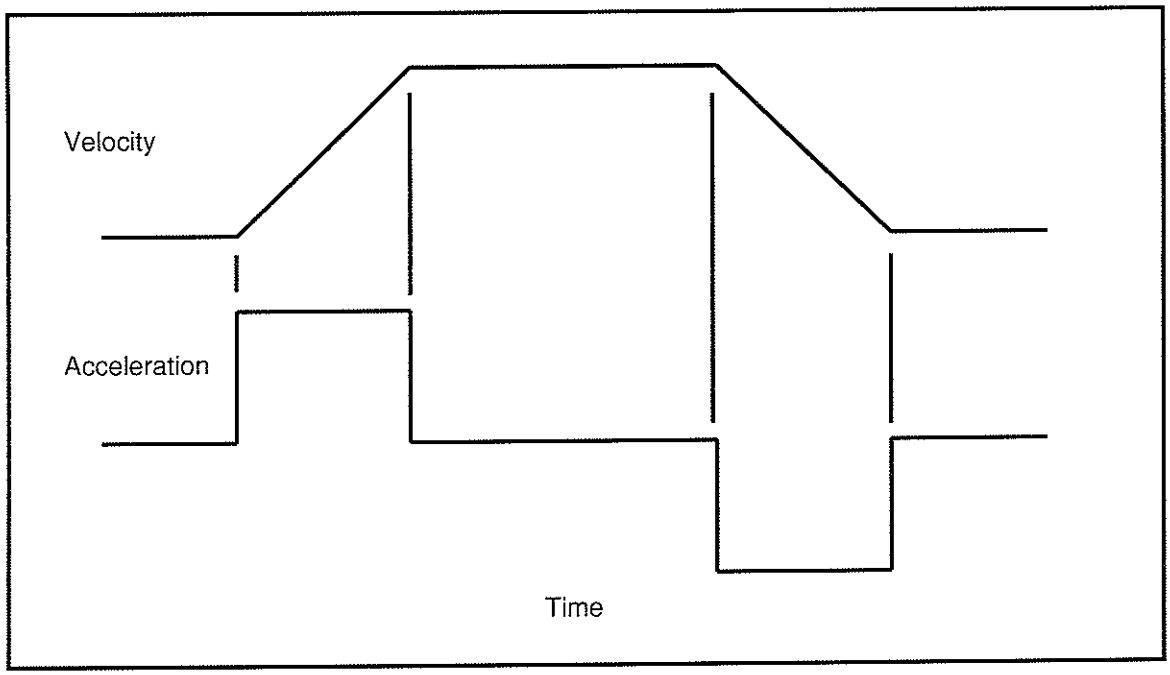


Figure 1-1 TYPICAL VELOCITY PROFILE

If the move parameters do not allow the motor to accelerate to the desired velocity in the desired distance, the SRX will automatically generate an optimum triangular velocity profile. It can also be commanded to accelerate to a velocity and hold that velocity until told to stop or change to a new velocity. It will then smoothly decelerate to a stop or accelerate/decelerate to the new velocity.

Several moves of this type may be chained together to provide a more complex pattern. The SRX is able to store up to 124 characters in an input character buffer, plus 200 commands and parameters in separate command queues for each axis, allowing several moves to be made without host intervention. A loop counter is provided to repeat desired sections of a complex move pattern. Loops may be nested up to four levels deep on all axes.

1.3. VELOCITY PROFILES

The SRX offers three options for ramping the device to speed. The traditional constant acceleration or linear velocity ramp (see Figure 1-1) is the default at power up or reset. The half sinusoid acceleration or half cosine velocity ramp (see Figure 1-3) is selected by the CN command. Since the acceleration is zero at the velocity inflection points, this offers very smooth operation. It is used in sensitive applications such as wafer handling on a vacuum chuck. The third option is a reverse ramp of acceleration or parabolic velocity curve (see Figure 1-2), which can be selected by the PN command. This ramp is commonly used to compensate for loss of motor torque at high speeds, i.e. since the acceleration is reduced at higher speeds the required forces are reduced proportionally. The parabola may be truncated to allow the user to select, under program control, the reduction in acceleration (force) appropriate for the application.

LINEAR RAMPS. The OMS controls generate a linear velocity ramp in real time, i.e. while the stage is in motion. There is no table building prior to the move and thus minimal latency. The controls will accelerate to the specified velocity and hold that speed until just enough move distance is left, then decelerate to a stop. If the move distance is too short to reach speed, a triangular velocity ramp will automatically be generated. The acceleration is a constant A_m and the velocity is then:

$$v = A_m t$$

A useful relationship is the distance required to accelerate at acceleration A_m to peak velocity V_p is:

$$s = \frac{V_p^2}{2A_m}$$

or the acceleration A_m required to accelerate to peak velocity V_p in distances s is:

$$A_m = \frac{V_p^2}{2s}$$

PARABOLIC RAMPS. The parabolic ramp is generated in a similar fashion except the acceleration is reduced as the stage accelerates to speed thus reducing the velocity slope as shown in Figure 1-2.

The acceleration follows the equation:

$$a = A_0 - A_0 \frac{t}{T_2}$$

and the velocity is then:

$$v = A_0 t - \frac{A_0 t^2}{2T_2}$$

and the distance traveled in the ramp is:

$$s = \frac{A_0 t^2}{2} - \frac{A_0 t^3}{6T_2}$$

where A_0 is the initial acceleration, t is time during the ramp and T_2 is total ramp time if the acceleration had reached zero. The parameter supplied with the PN command is 10 times the ratio $\frac{t}{T_2}$ which can take on values from 3 to 10, allowing the final acceleration to range from 70% to 10% respectively of the programmed or initial value. When a move is specified, the controls will fit the resulting velocity curve to the desired acceleration profile. This insures that the desired acceleration is always reached at the programmed velocity, as long as the move is long enough for the stage to reach the programmed speed. If the move is too short to reach the programmed speed the curve is truncated, causing the shape of the velocity curve to remain the same up to the velocity reached by the specific move. This is consistent with the desired result of compensating for loss of motor torque. Since the motor has not reached the programmed speed, less compensation is needed. The parabolic ramp mode may result in reduced move time at high speeds, since a larger acceleration may be used.

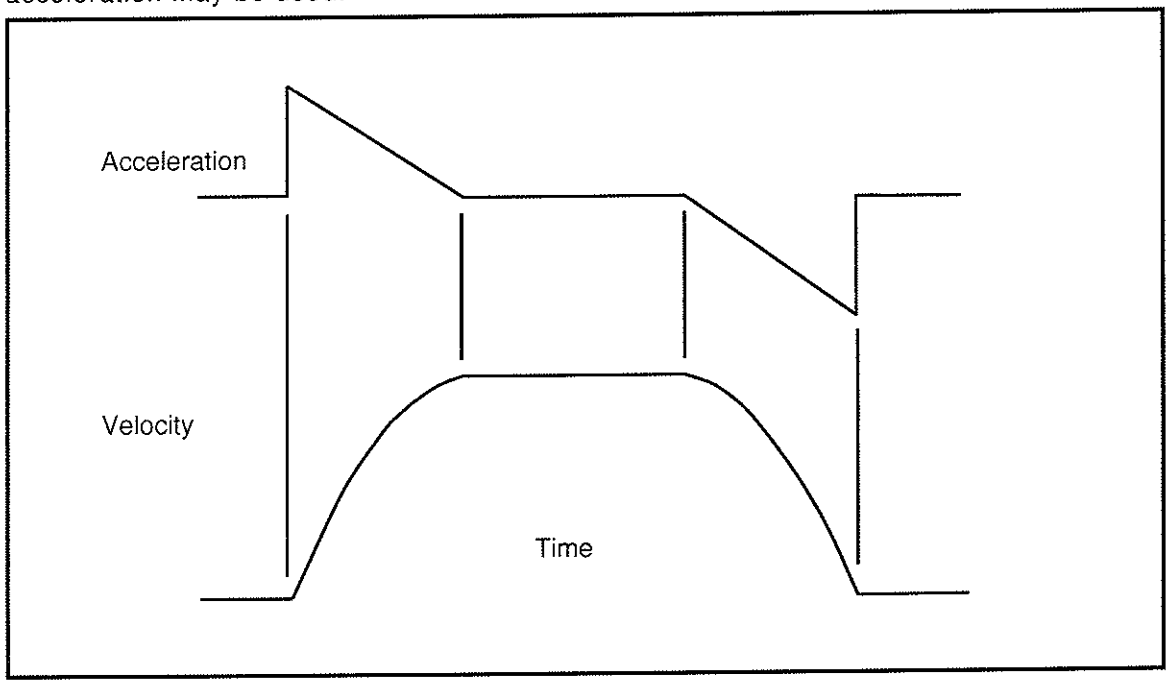


Figure 1-2 PARABOLIC VELOCITY PROFILE

COSINE RAMPS. The cosine ramps are generated in a similar fashion to the parabolic ramps, except the acceleration is:

$$a = A_m \sin \frac{2A_m}{V_p} t$$

and the velocity is then:

$$v = \frac{V_p}{2}(1 - \cos\frac{2A_m}{V_p}t)$$

and the distance traveled in the ramp is:

$$s = \frac{V_p}{2}t - \frac{V_p^2}{4A_m}\sin\frac{2A_m}{V_p}t$$

where V_p is the peak velocity, A_m is the peak acceleration. The distance needed to ramp up is then:

$$S_1 = \frac{\pi V_p^2}{4A_m}$$

and the time required to ramp up is:

$$T = \frac{\pi V_p}{2A_m} = \sqrt{\frac{\pi S_1}{A_m}}$$

and the peak velocity is:

$$V_p = \sqrt{\frac{4A_m S_1}{\pi}}$$

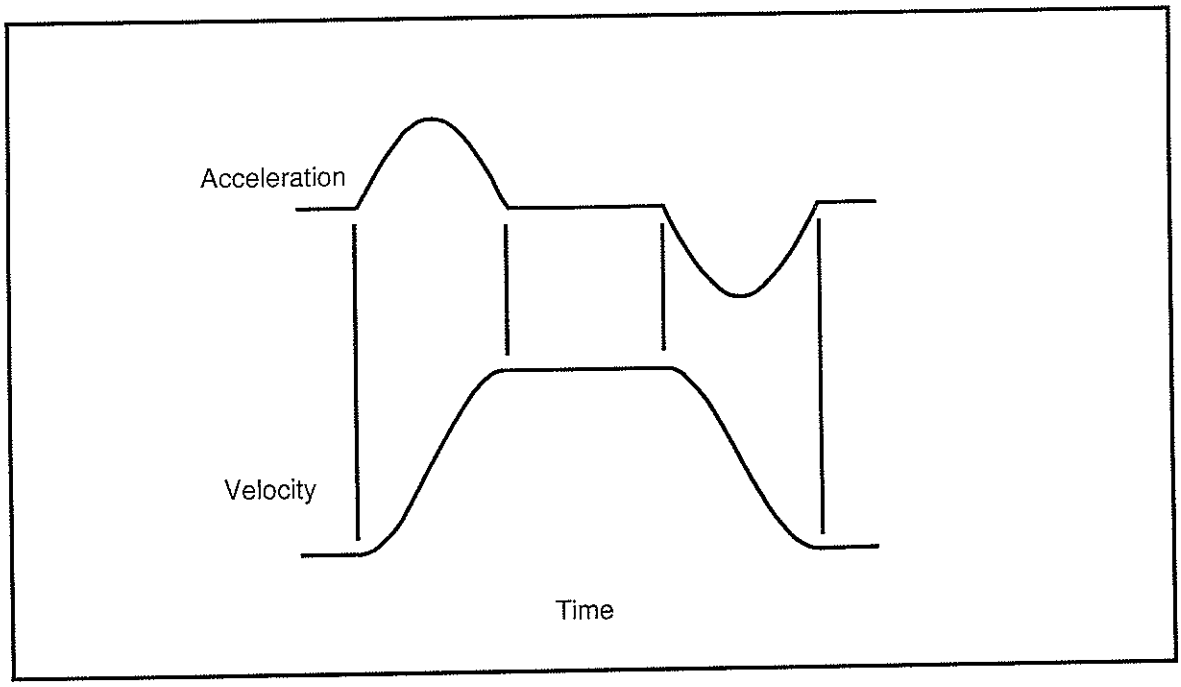


Figure 1-3 COSINE VELOCITY PROFILE

The cosine ramp requires $\frac{\pi}{2}$ times longer than a linear ramp to reach the same velocity when using the same peak acceleration.

Since the purpose of the cosine ramp is smooth operation, it is desirable to adjust the velocity parameters, such that the desired profile is achieved even when the stage does not reach the programmed speed, as opposed to truncating the curve as the parabolic modes do. The OMS controls look ahead to determine if the stage will be able to reach speed in the programmed move. If not, the acceleration curve will be adjusted such that the peak acceleration will be the programmed acceleration and the acceleration curve will be 360 degrees of a sine wave (see Figure 1-4).

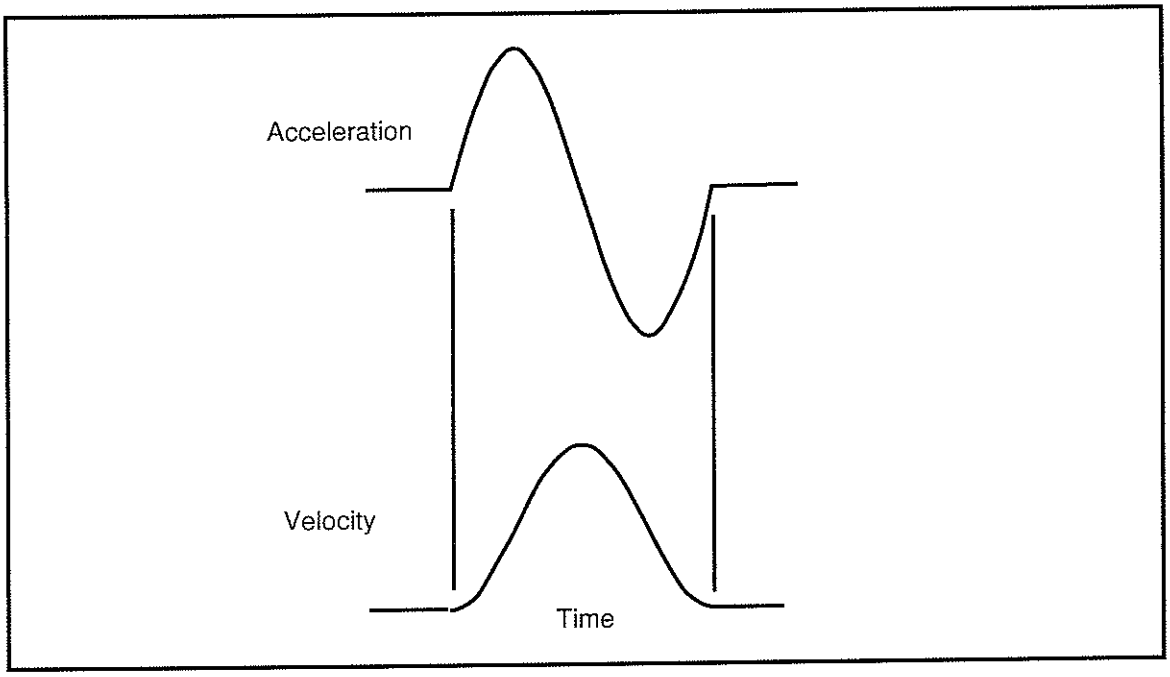


Figure 1-4 SHORT MOVE COSINE VELOCITY PROFILE

2. GETTING STARTED

2.1. INTRODUCTION

The SRX is a serial communication controller using EIA RS-232 or RS-422 communication standards. It can be configured as a stand alone PLC type controller for independent operation or can be operated on-line with a host computer for constant control and monitoring. It is equipped with a joy stick interface that can operate in either the on-line or stand alone configurations. The user definable I/O can be configured as either inputs or outputs. The default configuration is 16 inputs and 8 outputs. The configuration of the different features of the SRX may be altered to meet the individual users applications by changing a series of jumpers.

There are six blocks of square pin jumpers on the SRX board J28, J32, J22, J12, J4, J66, J81 and J17. See Figures 2-3, 2-4, and 2-8 for the location of the jumpers.

2.2. LIMIT POLARITY SELECTION

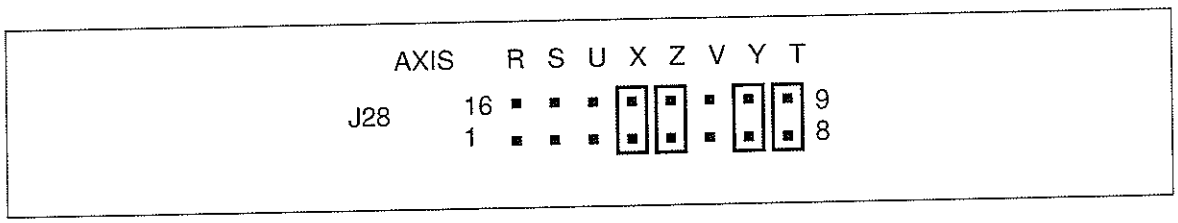


Figure 2-1 LIMIT SENSE JUMPERS FOR A FOUR AXIS BOARD

J28, which is near the lower right hand corner of the board, determines whether the limit inputs to an individual axis are active high or active low (see Figure 2-1). With the jumper in place, the associated axis will stop moving if the limit line, for the direction the axis is moving, is grounded. With the jumper removed, the axis will stop if the limit line is at +5VDC or open. These lines are internally pulled-up with a 2.2K ohm resistor to +5VDC so that only a switch closure is required to control them.

2.3. BOARD ADDRESS SELECTION

J32, located about three inches up from the lower left edge of the board (see Figure 2-2), serves two different functions. It selects the board serial address and also the configuration of the user definable I/O. Pins 1, 2 and 3 serve as A0, A1 and A2 respectively to select the board address. A jumper in place is a 1 while no jumper is a 0. Address 7, i.e. all three jumpers in place, is reserved for selection of the stand alone or PLC mode (see Section 4). Addresses 0 through 6 select the seven possible serial port addresses when multiple SRX boards are daisy chained on one serial line. Currently this function is not available.

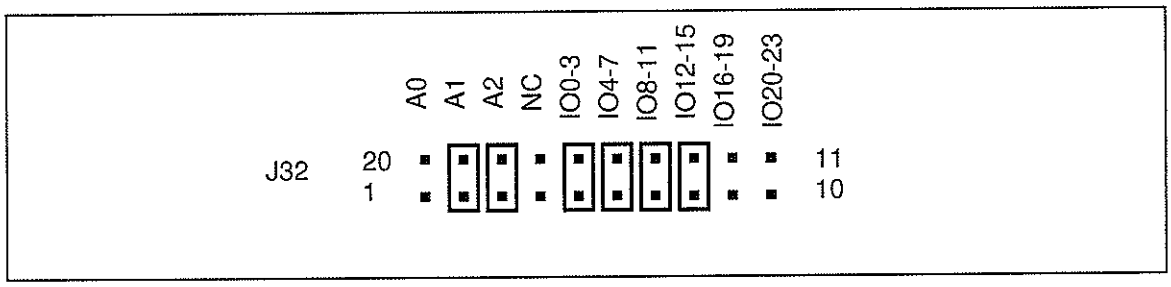


Figure 2-2 ADDRESS AND I/O SELECT JUMPERS (Default)

2.4. I/O CONFIGURATION

The user definable I/O is driven by TTL quad logic gates. A 7400 or open collector 7438 drives four outputs while a 74LS02 buffers four inputs. These ICs are installed in sockets for convenience and are located near the bottom middle of the board. They are U3, U4, U13, U14, U23 and U24. To change I/O bits 16 through 19 from outputs to inputs U24 must be changed from a 7400 to a 74LS02, J22 pins 5 through 8 should have jumpers added if pull-ups are desired and J32 pin pair 9 and 12 must have a jumper added to flag these pins as inputs to the SRX controller. The default configuration of the I/O has bits 0 through 15 as inputs and 16 through 23 as outputs. Figure 2-5 shows a typical input configuration while Figure 2-7 shows a typical output configuration.

Pin 4 of J32 is unused and should be ignored. Pins 5 through 10 are used to configure groups of 4 pins of user definable I/O as input or output. A jumper sets a group of four pins as inputs and no jumper sets them as outputs. Note that in order to change the configu-

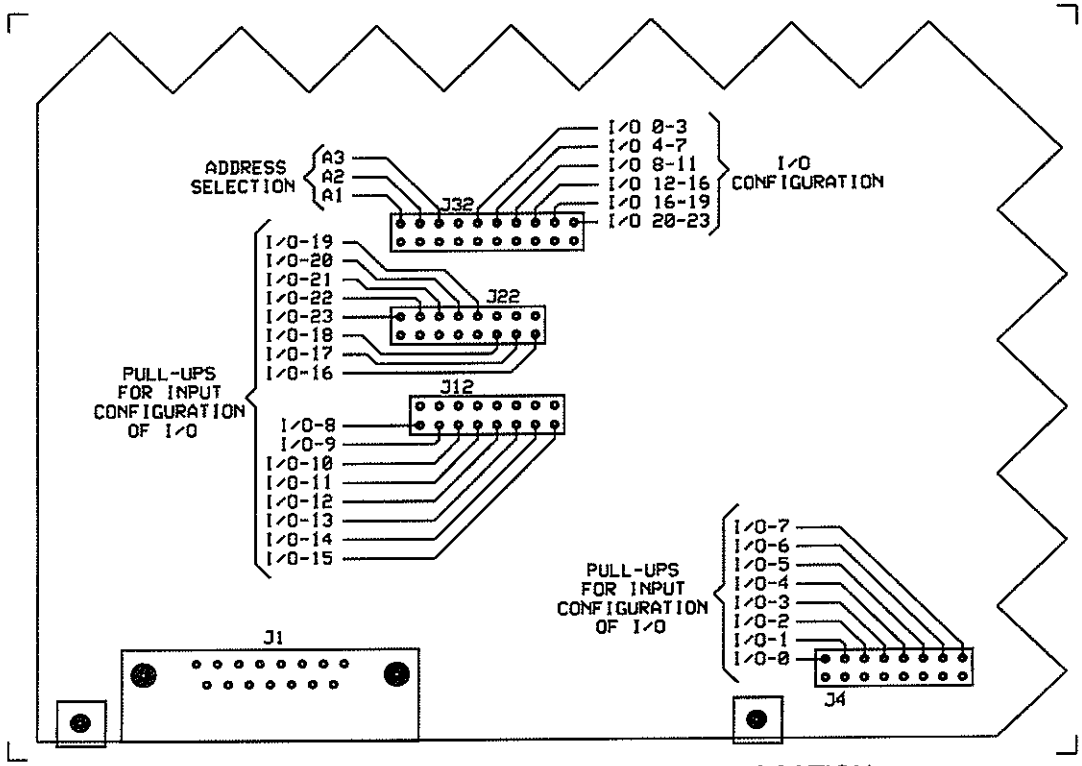


Figure 2-3 I/O CONFIGURATION JUMPER LOCATION

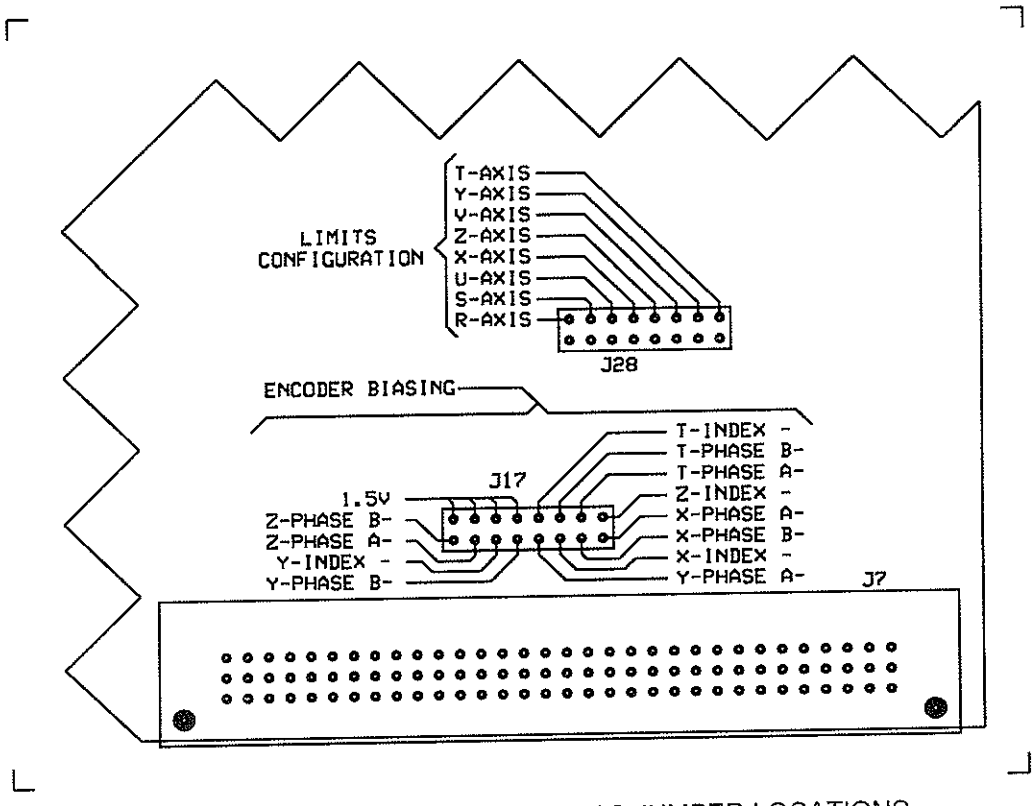


Figure 2-4 LIMITS AND ENCODER BIAS JUMPER LOCATIONS

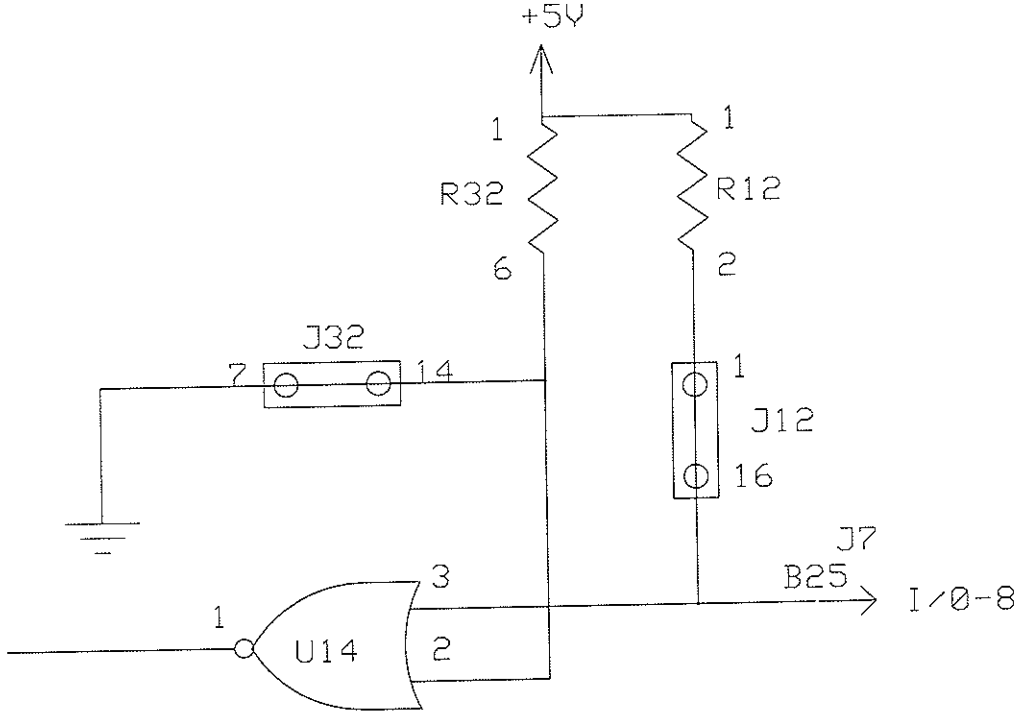


Figure 2-5 USER I/O INPUT CONFIGURATION

ration of the I/O, the corresponding IC, the pull-up jumper J22, J12 or J4 and the configuration jumper on J32 must all be changed.

J22, J12 and J4 are located on the lower left section of the board. They select pull-up resistors on the individual I/O when they are configured as inputs. When the I/O is configured as outputs the jumpers should be removed. Factory default has all 8 input connected to pull up resistors. In the stand alone or PLC modes, I/O bits 0 through 7 must be configured as inputs.

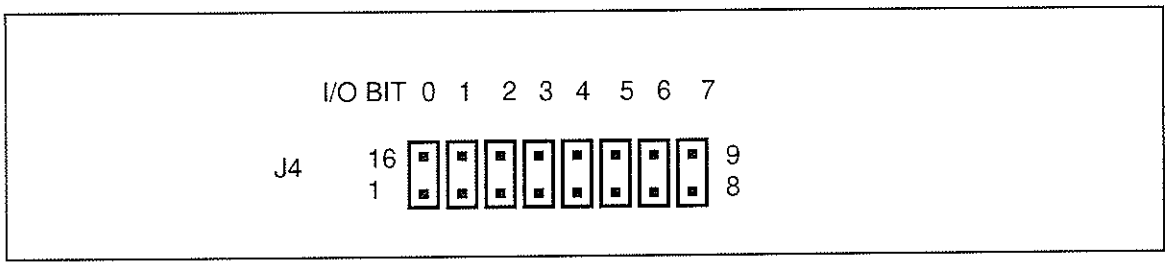


Figure 2-6 USER I/O PULL-UP JUMPERS (Default)

2.5. ENCODER BIAS CONFIGURATION

J17, located just above J7 in the lower right corner of the board, is used to bias unused encoder inputs. If single ended encoders are used, the unused line receiver inputs must be biased in the middle of the voltage swing of the active output. J17 provides a built-in bias supply of +1.5VDC for this purpose. Factory default is to have no jumpers on J17 (see Figure 2-4).

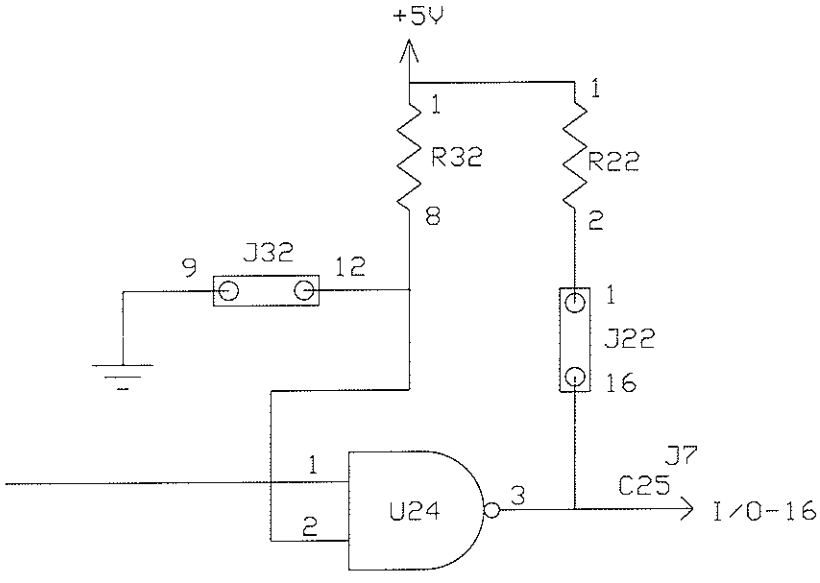


Figure 2-7 USER I/O OUTPUT CONFIGURATION

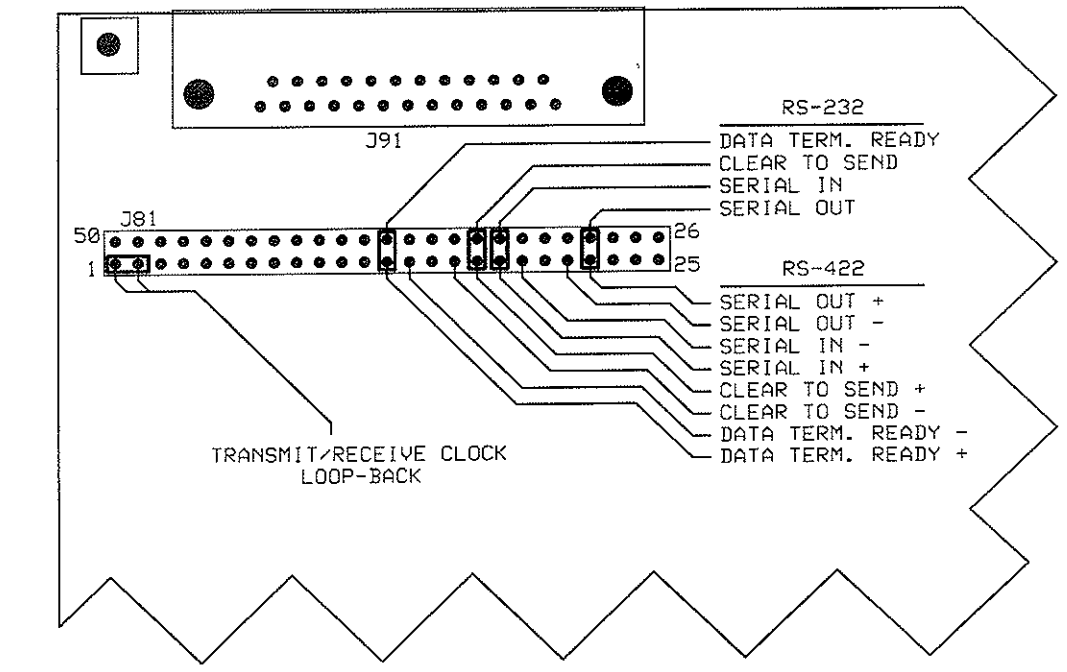


Figure 2-8 COMMUNICATION JUMPER LOCATION

2.6. SERIAL COMMUNICATION CONFIGURATION

The SRX utilizes the EIA RS-232 and RS-422 communication standards. J91 is the communication port of the SRX and is located in the upper left corner of the board. It is a 25 pin subminiature D connector and follows the EIA RS-232 circuit pin assignment. J81 is a jumper block that is used to route the signals to and from J91 allowing it to be configured as RS-232 or RS-422 and DTE or DCE. Refer to Figure 2-8 for the location of J81.

The SRX supports the Transmit Data (TxD), Received Data (RxD), Signal Ground and two control signals Clear to Send (CTS) and Data Terminal Ready (DTR). J81 connects these signals, except signal ground, to and from the connector J91 by the use of jumpers or wire wrap wire. The host computer or terminal is usually configured as a DTE. The default configuration for the SRX is a DCE so the cable connection to the DTE can be done straight through, i.e. no crossed wires. J91 may be configured as a null modem when the SRX is to be connected to another DCE such as a modem, by altering the jumper arrangement on J81. In either configuration, the control signals CTS (Clear to Send) and DTR (Data Terminal Ready) must be properly configured since they are used as ready signals by the SRX. These signals may also be crossed like a null modem when the SRX is to communicate with a DCE. Other control lines such as DSR (Data Set Ready) which are unused may be forced to the ready state, if desired, by wire wrapping to the +9VDC provided at pins 10, 11 or 12 of J81.

The SRX supports baud rates of 300, 600, 1200, 2400, 4800, 9600 and 19,200 bits per second. The default rate is 9600 baud and can be changed through the software command

SB# (Set Baud Rate). Once the baud rate has been changed on the SRX the communication software being used must then also be changed. An RR command (Ram Reset) will reinitialize all the on-board RAM and reset the baud rate to 9600, otherwise the last value set will remain in non-volatile memory even if power is removed from the board.

J81 provides for many different possible configurations for special applications. Most all of the connections of the Communication Port J91 are routed to the jumper block J81. The user is by no means limited to the communication examples in this manual. This manual does not attempt to explain all the possible communication configurations available. The intent is to give the user an understanding of the SRX communication interface and its possibilities and show some common examples.

The handshake control signals (CTS and DTR) may be looped back in applications where they are not supported by the host computer. The user must insure that the data sent to the SRX does not overrun its buffers if the handshaking is not used. Figure 2-8 shows the default configuration of the communication signals. For most applications this configuration will probably be sufficient. See Section 3 for additional information.

2.7. POWER SUPPLY REQUIREMENTS

The SRX is designed to operate from a +5VDC power supply. The SRX typical power consumption is 1.88 amps at +5VDC.

2.8. HARDWARE INSTALLATION

2.8.1. HOST CONTROL MODE

1. Configure jumpers as appropriate for the application. Use the default jumper configuration if possible before attempting any changes.

2. Connect the communication cable from the computer to the SRX connector J91.

3. Check to make sure the power supply is +5VDC and that it is fused (4 amp maximum) before connection to the SRX.

4. Mount the SRX board on stand-offs to insure it is not touching any conductive materials.

5. Connect the power supply to J92 or J95, double checking the polarity, and then turn it on. On J92, the +5VDC connects to the pin closest to the LED, ground connects to the center pin and the third pin is left unconnected. Refer to Figure 2-9.

6. Observe the red LED (L92) on the SRX board. It should be on at power up then turn off about a second after. Each time the reset button is pressed, the LED should light for about 1 second. The LED shows when the board is in a reset or fault condition. If it remains on continuously, the board is malfunctioning. If the LED does not turn off, examine the SRX for debris on the front and back and look for loose or damaged parts.

7. Use a serial communication program that can send ASCII characters to the SRX board at 9600 baud rate.

8. Send the command WY followed by a carriage return. This asks the SRX "who are you". The SRX should return the model type and firmware version number (i.e. SRX ver 1.64-8).

9. If reconfiguration of the communication port is desired, disconnect the power and refer to Section 3. Then repeat steps 1 through 8 above. If there are any problems contact Oregon Micro Systems, Inc. for assistance.

10. Connect motor drivers to J7. See Section 5 for details. Double check all connections.

11. Apply driver power.

12. Send a command such as JG1000, to an axis connected to a driver to cause the motor to move.

13. Check that the motor is moving and is going the desired direction.

14. If you have any problems double check your connections.

15. Call Oregon Micro Systems, Inc. if you need further assistance.

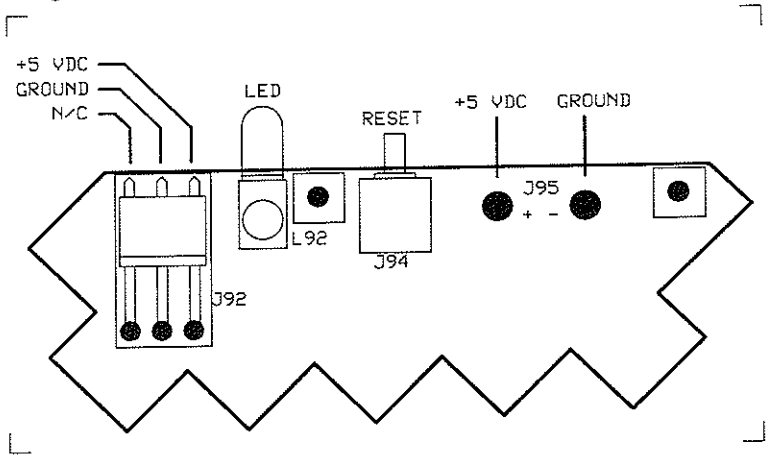


Figure 2-9 POWER SUPPLY CONNECTIONS

2.8.2. STAND ALONE SETUP AND OPERATION

1. Follow the Hardware Installation procedure so you can talk to the board through the serial port.

2. Go to the AA (Axes All) mode and set the velocity and acceleration for all axes.

3. Type JY (Joy Stick Mode) to enter the joy stick mode and then type a JZ (Joy Stick Zero) to electrically zero the joy stick to the SRX.

4. Type JD5 (Joy Stick Deadband). This sets the joy stick deadband to 5, though any value from 0 to 10 can be entered.

5. Set the board address to 7 by placing jumpers on pin pairs 1-20, 2-19, 3-18 of J32.

6. With the board set to address 7 press the reset button, turn power off then on, or send the RS command to the board. This will cause the board to go into the stand alone mode. DO NOT send a RAM Reset (RR) or the board will loose acceleration, velocity, and other values previously defined.

7. Connect PLC controller to PLC port of the SRX. See Section 4 for details.

8. Connect motor drivers to J7. See Section 5 for details.

9. Set the left switch on the joy stick module to the down position for the RUN mode.

10. Execute an HA command by inputting 120 decimal into the PLC control port and asserting the strobe. This will cause the SRX to find its motor's home position.

11. Set the left switch on the joy stick module to the up position for the TEACH mode and set the center switch so the near side is down to select the X-Y axis pair. Move the joy stick to test the motors. Their movement should follow the joy stick.

12. If the motors do not move, check all connections to the motors and the SRX board. Check the connection to the joy stick module. Check the setting of the left switch to see that its side closest to the user is in the up position and check the center switch to see that it is set for the desired axis pair.

13. Using the joy stick, move each axis pair to the desired locations, moving the center switch to select the axis pair to move.

14. Press the right switch momentarily to teach that position to the SRX memory.

15. Repeat steps 13 and 14 until all positions have been taught. Positions may be manually entered by using the PE command.

16. Move the left switch to the down position to enter the run mode.

17. With all inputs to the parallel port open, assert the strobe (I/O bit 7 to ground). The stage should move to the first position (position 0) taught in steps 13 and 14. If the stage does not move to the first position, double check the left switch to see that it is in the down position.

18. Repeat step 14 for all positions that have been taught. If problems arise, repeat the process from step 1. If problems persist, contact Oregon Micro Systems, Inc. for assistance.

2.9. MOTOR CONTROL CONNECTOR

The motor control connector (J7) on the SRX board consists of three rows of square pins, each row has 32 pins for a total of 96 pins. Twenty four of the pins are user definable I/O which can be used to synchronize motor movement to external events. One pin provides an external reset input to the board.

The other pins can be considered as 8 logical sets of 6 pins where each set is used for an individual axis. Pins for +5VDC and ground are also provided for the user's convenience. If your board is not an 8 axis board or a 4 axis board with encoder feedback then some of the sets of pins will not be used. The 6 pins of an axis set are: Step Output, Auxiliary

Output, Direction Output, Negative Limit Switch Input, Home Switch Input and Positive Limit Switch Input. See Section 5 for a detailed description of the connector.

Boards with the Encoder Option use two or four sets of 6 pins as encoder input for the X, Y, Z and T axes. The 6 pins of an encoder pin set are: Index +, Phase A -, Phase A +, Index -, Phase B -, Phase B +. See Section 6 for a detailed description of the connector. Connections to the +5VDC logic supply and ground are also provided for the user's convenience in providing power to the encoder.

3. SERIAL INTERFACE

3.1. SERIAL COMMUNICATION

The SRX utilizes the EIA RS-232 and RS-422 communication standards. J91 is the communication port of the SRX and is located in the upper left corner of the board. It is a 25 pin subminiature D connector and follows the EIA RS-232 circuit pin assignment. J81 is a jumper block that is used to configure the signals to and from J91. Refer to Table 3-2 for details about J81.

Table 3-1 RS-232 CONNECTOR STANDARD

PIN	SYMB	SIGNAL DESCRIPTION	SIGNAL SOURCE
1	AA	Protective Ground	
2	BA	Transmit Data	Terminal (DTE)
3	BB	Receive Data	Modem (DCE)
4	CA	Request To Send	Terminal (DTE)
5	CB	Clear To Send	Modem (DCE)
6	CC	Data Set Ready	Modem (DCE)
7	AB	Signal Ground	
8	CF	Receive Line Signal Detect	Modem (DCE)
9		Reserved	
10		Reserved	
11		Unassigned	
12	SCF	Secondary Receive Line Signal Detect	Modem (DCE)
13	SCB	Secondary Clear To Send	Modem (DCE)
14	SBA	Secondary Transmit Data	Terminal (DTE)
15	DB	Transmit Signal Element Timing	Modem (DCE)
16	SBB	Secondary Receive Data	Modem (DCE)
17	DD	Receiver Signal Element Timing	Modem (DCE)
18		Unassigned	
19	SCA	Secondary Request To Send	Terminal (DTE)
20	CD	Data Terminal Ready	Terminal (DTE)
21	CG	Signal Quality	Modem (DCE)
22	CE	Ring Indicator	Modem (DCE)
23	CH/CI	Data Signal Rate Select	T/M
24	DA	Transit Signal Element Timing	Terminal (DTE)
25		Unassigned	

Table 3-2 SRX SERIAL INTERFACE CONNECTOR PIN LIST

J91 PIN #	SIGNAL DESCRIPTION	J81 PIN #		SRX DESCRIPTION
		to J91	to SRX	
1	Protective ground	26	25	+1.5VDC Bias Supply
7	SRX Signal Ground	27	24	Reserved
	No Connection	28	23	Reserved
3	Receive Data	29	22	+ Transmit Data Out
9	Reserved	30	21	- Transmit Data Out
	No Connection	31	20	+1.5VDC Bias Supply
10	Reserved	32	19	- Receive Data In
2	Transmit Data	33	18	+ Receive Data In
5	Clear To Send	34	17	+ Clear To Send
11	Unassigned	35	16	- Clear To Send
	No Connection	36	15	+1.5VDC Bias Supply
12	Sec. Rec. Line Sig. Det.	37	14	- Data Terminal Ready
20	Data Terminal Ready	38	13	+ Data Terminal Ready
4	Request To Send	39	12	+9VDC
6	Data Set Ready	40	11	+9VDC
8	Rec. Line Sig. Det.	41	10	+9VDC
14	Secondary Transmit Data	42	9	+1.5VDC Bias Supply
16	Secondary Receive Data	43	8	- Receive Clock
17	Rec. Sig. Elem. Timing	44	7	+ Receive Clock
15	Trans. Sig. Elem. Timing	45	6	+ Transmit Clock
24	Trans. Sig. Elem. Timing	46	5	- Transmit Clock
21	Signal Quality	47	4	Reserved
22	Ring Indicator	48	3	Reserved
23	Data Signal Rate Select	49	2	SRX Transmit Clock
	Receive Clock Enable	50	1	SRX Receive Clock
13	No Connection			
18	No Connection			
19	No Connection			
25	No Connection			

The SRX can be configured as a DCE (Data Communication Equipment) or DTE (Data Terminal Equipment) device. The host computer or terminal is usually configured as a DTE. The default configuration for the SRX is a DCE allowing a cable connection that is straight through, i.e. no crossed wires. J91 may be configured as a null modem when the SRX is to be connected to another DCE such as a modem. In either configuration, the control signals CTS (Clear To Send) and DTR (Data Terminal Ready) must be properly configured since they are used as ready signals by the SRX. These signals may also be crossed like a null modem when the SRX is to communicate with a DCE. Other signals such as DSR (Data Set Ready) may be forced to the ready state, if desired, by wire wrapping to the +9VDC provided at pins 10, 11, or 12 of J81.

The SRX supports baud rates of 300, 600, 1200, 2400, 4800, 9600 and 19,200 bits per second. The default rate is 9600 and can be changed through the software command SB# (Set Baud Rate). Once the baud rate has been changed on the SRX, the communication software being used must then also be changed. An RR command (Ram Reset) will reinitialize all the on-board RAM and reset the baud rate to 9600.

J81 provides for many different possible configurations for special applications. Most all of the connections of the Communication Port J91 are routed to the jumper block J81. The user is by no means limited to the communication examples in this manual. This manual does not attempt to explain all the possible communication configurations available. The intent is to give the user an understanding of the SRX communication interface and its possibilities and show some common examples.

3.2. RS-232

Table 3-1 shows the EIA standard RS-232 signal pin assignments. The SRX supports the Transmit Data (TxD), Received Data (RxD), Signal Ground and two control signals Clear to Send (CTS) and Data Terminal Ready (DTR). J81 connects these signals, except signal ground, to and from the connector J91 by the use of jumpers or wire wrap wire. This provides the ability to loop back the control signals or redirect signals to different pins on J91 for special applications. Note that control signals and handshaking signals are the same as the modem control signals CTS, RTS, DTR, and DSR, etc.

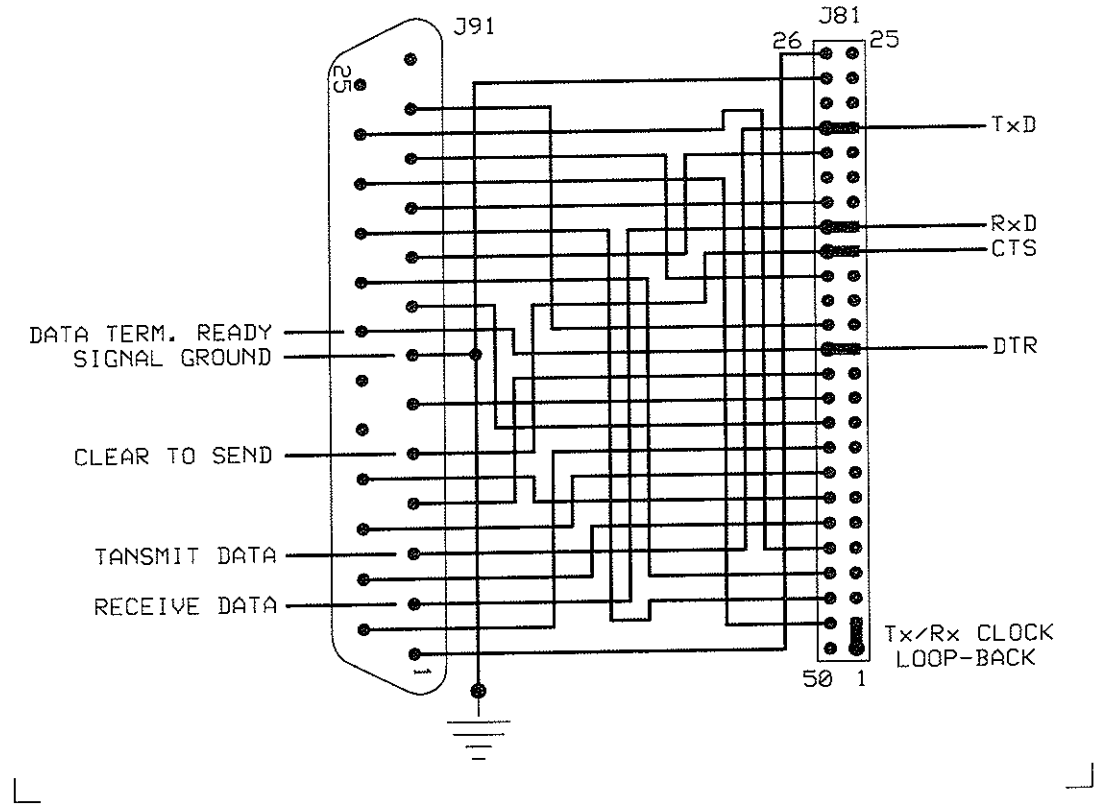


Figure 3-1 RS-232 DEFAULT JUMPERS

Table 3-3 RS-422 CONNECTOR STANDARD

PIN	SYMB	SIGNAL DESCRIPTION	SIGNAL SOURCE
1		Shield	
2	SI	Signal Rate Indicator	Modem
3		Spare	
4	SD+	Send Data	Terminal
5	ST+	Send Timing	Modem
6	RD+	Receive Data	Modem
7	RS+	Request To Send	Terminal
8	RT+	Receive Timing	Modem
9	CS+	Clear To Send	Modem
10	LL	Local Loopback	Terminal
11	DM+	Data Mode	Modem
12	TR+	Terminal Ready	Terminal
13	RR+	Receiver Ready	Modem
14	RL	Remote Loopback	Terminal
15	IC	Incoming Call	Modem
16	SR/SR	Select Frequency/Signal Rate Select	Terminal
17	TT+	Terminal Timing	Terminal
18	TM	Test Mode	Modem
19	SG	Signal Ground	Terminal
20	RC	Receive Common	Modem
21		Spare	
22	SD-	Send Data	Terminal
23	ST-	Send Timing	Modem
24	RD-	Receive Data	Modem
25	RS-	Request To Send	Terminal
26	RT-	Receive Timing	Modem
27	CS-	Clear To Send	Modem
28	IS	Terminal In Service	Terminal
29	DM-	Data Mode	Modem
30	TR-	Terminal Ready	Terminal
31	RR-	Receiver Ready	Modem
32	SS	Select Standby	Terminal
33	SQ	Signal Quality	Modem
34	NS	New Signal	Terminal
35	TT-	Terminal Timing	Terminal
36	SB	Standby Indicator	Modem
37	SC	Send Common	Terminal

For example, some host computers will require the DSR signal to be true before they will transmit. In this case, the user could bias the signal to +9VDC by placing a wire-wrap on pins 40 and 11 of J81.

The handshake signals (CTS and DTR) may be looped back in applications where they are not supported by the host computer. It is up to the programmer, then, to insure that commands to the board do not overrun the SRX's buffer, since the SRX will no longer be able to signal the host when its buffer is full. Figure 3-1 shows the default configuration of the communication signals. For most applications this configuration will probably be sufficient.

3.3. RS-422

The SRX utilizes the MC3486 quad line receiver and MC3487 quad line driver to comply with the EIA standard for balanced asynchronous RS-422 communication. Note that the RS-232 driver IC must be removed before installing the 3486 and 3487 ICs since they both drive the same signal lines. The following differential signals are supported: Transmit Data, Receive Data, Clear to Send, Data Terminal Ready, Transmit Clock, and Receive Clock. The default utilizes these signals, looping-back the Transmit and Receive Clocks. Figure 3-2 shows the default RS-422 communication configuration.

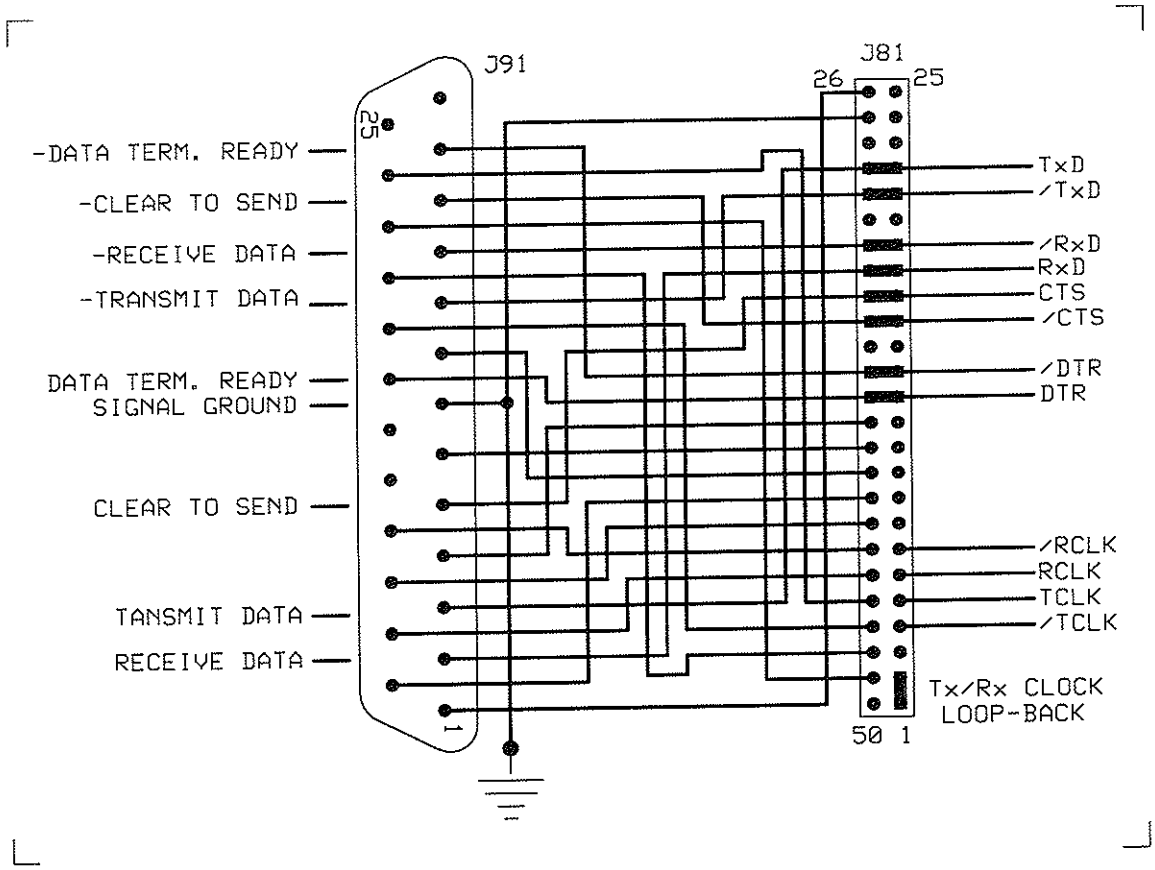


Figure 3-2 RS-422 DEFAULT JUMPERS

In most applications the transmit (TxC) and receive (RxC) clocks are not used and should be looped back as in Figure 3-2. If the SRX is configured as a DCE, the clock signals would be jumpered straight across the jumper block.

The RS-422 interface shares the same J91 and J81 configuration jumper as the RS-232. This gives the RS-422 the same ability of looping back the control signals. It also provides the same ability to redirect the communication port J91 pin assignment for special applications. Refer to Figure 3-2 for RS-422 configurations. Unused inputs should be tied to +5VDC or ground, if not used.

3.4. CONVERTING FROM RS-232 TO RS-422

Remove power first. The RS-232 and RS-422 communication protocols utilize the same signal lines, therefore the RS-232 driver and receiver IC U81 must be removed from its socket when converting to RS-422. After removing U81, install an MC3487 or equivalent in the socket at U71 and an MC3486 or equivalent in the socket at U72. Then referring to Figure 3-2 and Table 3-3, install the necessary jumpers or wire wraps for the appropriate communication configuration. The recommended or default configuration is shown. The SRX is now ready for communication with the RS-422 device.

4.
PLC AND JOY STICK INTERFACE.

4.1. INTRODUCTION

The SRX board, besides being able to be controlled through the serial port, can be directed to learn and perform moves by using the combination of the parallel (PLC) interface and the joy stick module.

The parallel interface uses nine of the user I/O bits on the output connector J7. Using this interface, the user can easily command the SRX board via a Programmable Logic Controller (PLC) or with several switches that connect the desired PLC lines to ground.

The joy stick is used to position the desired axes into the correct position. The position is stored (taught) into the SRX's non-volatile memory when the right-most switch on the joy stick is pressed. The SRX can then be commanded at some future time to move to the taught position by placing that position number on the PLC port and toggling the strobe line.

4.2. PLC PORT

The SRX board is directed through the PLC port by a series of simple commands encoded as a binary word of seven active low input bits. An eighth input bit provides an active low strobe to the SRX board, signaling when a valid bit pattern is asserted on the port. A ninth bit, output from the SRX, provides an active low BUSY signal to the PLC controller, showing that the previously programmed move or macro is still executing.

If the PLC mode is not needed these I/O bits can be used the same as the other I/O on the board. If the PLC mode is used, however, the 9 bits should not be used as I/O or the board may produce unexpected results.

Table 4-1 shows which physical pins on J7 of the SRX board are used by the PLC port.

The I/O pins are also listed in tables in Sections 5 and 6.

Table 4-1 PLC INTERFACE CONNECTIONS

J7 PIN	I/O BIT	PLC VALUE
A25	0	1
A26	1	2
A27	2	4
A28	3	8
A29	4	16
A30	5	32
A31	6	64
A32	7	Strobe
C32	23	Busy

4.3. STORING POSITIONS IN MEMORY

There are three ways a position can be entered into the SRX's 100 position memory:

(1): Using the serial port to send the PE#,# command to the board which allows all 8 axes to be utilized.

(2): Enter the joy stick mode from the serial port with the JY command and use the joy stick to move the axes to the desired position, switch the left joy stick switch to the teach position and toggle the right joy stick switch. Each time the right switch is pressed, the current positions that would be shown by the RP command for all axes are copied to the current position memory location. This also causes the position memory location pointer to increment. The result is that the first time the switch is toggled position 0 is taught, the second time position 1 is taught, the third time teaches position 2, etc. The position number taught may be changed by using the JN command.

(3): From the serial port, set up the axes with the desired velocity, acceleration and other parameters, enter the joy stick mode and zero the joy stick and set its deadband. Set jumper J32 for address 7 by putting a jumper across pin pairs 1-20, 2-19, 3-18 and either turn the power to the board off then back on, send an RS command or press the reset button on the board. Do NOT send an RR command or the velocities and accelerations you just programmed will be lost. Send an HA command (120) to the PLC port. This will cause the motors to find their home position. Put the left joy stick switch in the teach position. Move the axes to the desired positions using the joy stick. Toggle the right joy stick switch, this will cause the current positions to be stored into position 0, move the axes to the next position and toggle the right switch, this will teach position 1, the third time will teach position

2, etc. The next position number taught can be changed at any time by placing the position number on the PLC port and toggling the strobe.

4.4. MOVING TO POSITIONS IN MEMORY

There are three ways to cause the SRX to make the moves to the taught positions, the moves will be linear ones as performed by the MT command:

(1): Use the serial port to send the PT# command to the board.

(2): Use the serial port to set the SRX board into the joy stick run mode with the JR command, switch the left joy stick switch to run, then enter the position memory location as a value in binary format on the PLC port. The move will start when the STROBE line (I/O bit 7) is toggled low.

(3): Set the desired velocity and acceleration values through the serial port. Set jumper J32 for address 7 by putting a jumper across pin pairs 1-20, 2-19, and 3-18. Disconnect the joy stick or put the left switch in the run mode. Put the PLC value for the desired memory location onto the PLC port. When the strobe line is toggled the move will start. The BUSY line (I/O bit 23) will go low until the move is ended.

4.5. PLC COMMANDS

There are 128 different possible PLC commands that can be sent to the PLC port on the SRX board. Some of them are invalid.

The position of the RUN/TEACH switch on the joy stick also changes the effects of the commands. Tables 4-2 and 4-3 show the PLC commands and their PLC code.

Table 4-2 PLC RUN MODE COMMANDS

COMMAND	FUNCTION
0-99	Execute PT command for that position
100-119	Execute MX command for macros 0-19
120	Execute HA command
121	Execute HJ command
122-126	Not valid
127	Execute KL command

Table 4-3 PLC TEACH MODE COMMANDS

COMMAND	FUNCTION
0-99	Execute JN command for that position
100-127	Not valid

See Section 7 for a complete explanation of the SRX commands listed in this section.

4.6. JOY STICK OPERATION

The joy stick interface on the SRX board, as previously mentioned, allows the user to easily move up to six axes of the SRX board to a desired location.

There are four controls on the joy stick and one 15 pin D type connector. The controls only have an effect if the SRX board is in the JY or JT modes or if the board was powered up with its address jumper J32 set for address 7, causing the board to be in the stand alone mode.

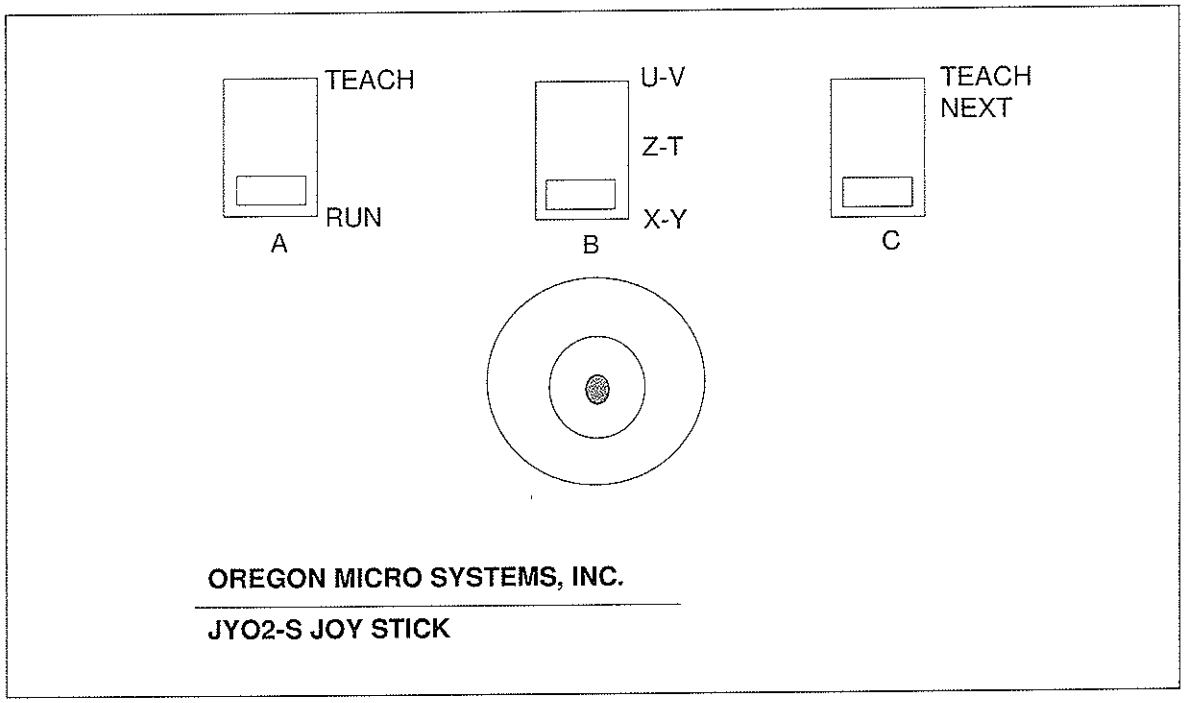


Figure 4-1 JY02-S JOY STICK

Switch A, the left most toggle switch, controls whether the SRX board is in the TEACH or RUN modes. With the end of the switch closest to the user pressed down, the board is in the RUN mode and with the far side pressed down, the board is in the TEACH mode.

Switch B, the center three position toggle switch, selects which axis pair the joy stick will control. With the end of the switch closest to the user pressed down, the joy stick will

control the X-Y axis pair. With the switch in the center position, the joy stick controls the Z-T axis pair. With the far side of the switch pressed down, the joy stick operates the U-V axis pair.

Switch C, the right most momentary switch, causes the current position of all axes to be stored into the currently selected Position Memory location if the RUN/TEACH switch is in the TEACH POSITION.

The fourth control is the joy stick itself. The stick is attached internally to a pair of potentiometers. These produce a varying voltage that is interpreted by hardware and software to set the speed and direction of the axis pair currently selected by the center switch.

The actual speed produced using the joy stick is a result of the velocity and acceleration defined for the axis pair and the distance from center the joy stick has moved. The default velocity and acceleration of the SRX produce a pulse stream that may not have the resolution of movement desired. If that is the case, define new velocities and accelerations that fit your application better.

The 15 pin D connector on the rear of the joy stick module provides a signal path to the SRX board. Table 4-4 shows a description of the pins of this connector.

Table 4-4 JOY STICK MODULE CONNECTOR

PIN	FUNCTION
1	Unused
2	Unused
3	Unused
4	Unused
5	Unused
6	Unused
7	Vertical Movement Analog Output
8	Horizontal Movement Analog Output
9	Ground
10	Ground
11	Left Switch Output
12	Center Switch Output
13	Center Switch Output
14	Right Switch Output
15	+5VDC Input

4.7. JOY STICK COMMANDS

There are several commands that are used with the joy stick. Among them are:

JY. Which causes the SRX to enter the joy stick mode.

JZ. Which causes the SRX to view the center position of the joy stick as the position where both axes have a velocity of 0.

JD. Which adjusts the sensitivity of the center position of the joy stick.

JN. Which allows the user to define which Position Memory position will be taught next.

JT. Which allows the PLC port to select which Position Memory position will be taught.

JR. Which allows the PLC port to select which command or position to execute.

See Section 7 for a complete explanation of the SRX commands listed in this section.

5. DRIVER INTERFACE

5.1. OUTPUT CONNECTIONS

Table 5-1 lists the input and output interface signals available at output connector J7 on each SRX board. The connector is a DIN type of 3 rows of 32 pins on 0.1 inch centers. The connector pin assignment is shown as viewed looking into the connector on the board. The mating connector is an AMP, Inc. part number 532509-1 with a 532508-1 hood and strain relief and crimp contacts 530151-5.

Refer to Section 6 for pin assignments on SRX boards with the encoder option.

A separate 4-conductor shielded cable should be used for each axis for connections to its associated driver module and must be limited to 50 feet. A connection to the SRX +5VDC power is provided for each axis to supply power to the emitter diode within an opto-isolated motor driver module, such as the MH10. This allows the use of such drivers without the need for an external power supply.

CAUTION:

> This power supply connection must not be connected to any other supply or used for any other purpose or damage may result to the host computer or the SRX or both.

A ground connection is provided for each axis for convenience in connecting up the system. The SRX is supplied with 7406 open collector TTL drivers as standard. These parts are in sockets and can be replaced with 7404 totem pole drivers for driver modules which do not have opto-isolated inputs. Each device handles the step, direction and auxiliary output for two axes. U29 on the board handles the X and Y axes, U19 handles the Z and T axes, U17 is for the U and V axes while U26 handles the R and S axes. The cable shields should be connected to the appropriate ground pins as shown in Table 5-1, and left open at the driver end when used with opto-isolated loads to avoid ground loops and ensure isolation.

5.2. MULTI-AXIS SYNCHRONIZATION

Each SRX has provision for synchronizing several SRX boards to drive larger systems. Systems requiring more than 8 axes and thus more than one SRX can be synchronized by connecting an auxiliary output on one board to the user definable input on the other board. The boards can signal each other at the appropriate place in the command stream without interrupting the host computer. Synchronization can be accomplished with other devices as well.

Table 5-1 J7 INPUT AND OUTPUT PIN ASSIGNMENTS

PIN	ROW C FUNCTION	ROW B FUNCTION	ROW A FUNCTION
1	Ground	NC	+5VDC
2	X Auxiliary Output	X Direction Output	X Step Output
3	X Home Switch	X Positive Limit Switch	X Negative Limit Switch
4	Y Auxiliary Output	Y Direction Output	Y Step Output
5	Y Home Switch	Y Positive Limit Switch	Y Negative Limit Switch
6	Ground	NC	+5VDC
7	Z Auxiliary Output	Z Direction Output	Z Step Output
8	Z Home Switch	Z Positive Limit Switch	Z Negative Limit Switch
9	T Auxiliary Output	T Direction Output	T Step Output
10	T Home Switch	T Positive Limit Switch	T Negative Limit Switch
11	NC	NC	NC
12	Ground	NC	+5VDC
13	U Auxiliary Output	U Direction Output	U Step Output
14	U Home Switch	U Positive Limit Switch	U Negative Limit Switch
15	V Auxiliary Output	V Direction Output	V Step Output
16	V Home Switch	V Positive Limit Switch	V Negative Limit Switch
17	Ground	NC	+5VDC
18	R Auxiliary Output	R Direction Output	R Step Output
19	R Home Switch	R Positive Limit Switch	R Negative Limit Switch
20	S Auxiliary Output	S Direction Output	S Step Output
21	S Home Switch	S Positive Limit Switch	S Negative Limit Switch
22	NC	NC	NC
23	NC	NC	NC
24	Ground	Reset Input	+5VDC
25	User I/O Bit 16	User I/O Bit 8	User I/O Bit 0*
26	User I/O Bit 17	User I/O Bit 9	User I/O Bit 1*
27	User I/O Bit 18	User I/O Bit 10	User I/O Bit 2*
28	User I/O Bit 19	User I/O Bit 11	User I/O Bit 3*
29	User I/O Bit 20	User I/O Bit 12	User I/O Bit 4*
30	User I/O Bit 21	User I/O Bit 13	User I/O Bit 5*
31	User I/O Bit 22	User I/O Bit 14	User I/O Bit 6*
32	User I/O Bit 23*	User I/O Bit 15	User I/O Bit 7*

* = PLC Control Port

5.3. LIMIT AND HOME LINES

The limit and home lines can be activated using mechanical switches using contact closures or other suitable active switches, such as a hall effect switch or opto-isolator, that connect the line to ground. The limit switch closure will stop the associated pulse stream if the motor travels beyond its allowable limits and trips the switch. The home switch provides a means to synchronize the motor controller with the load at some home or reference position. The home switch, when used with the software HM command, will

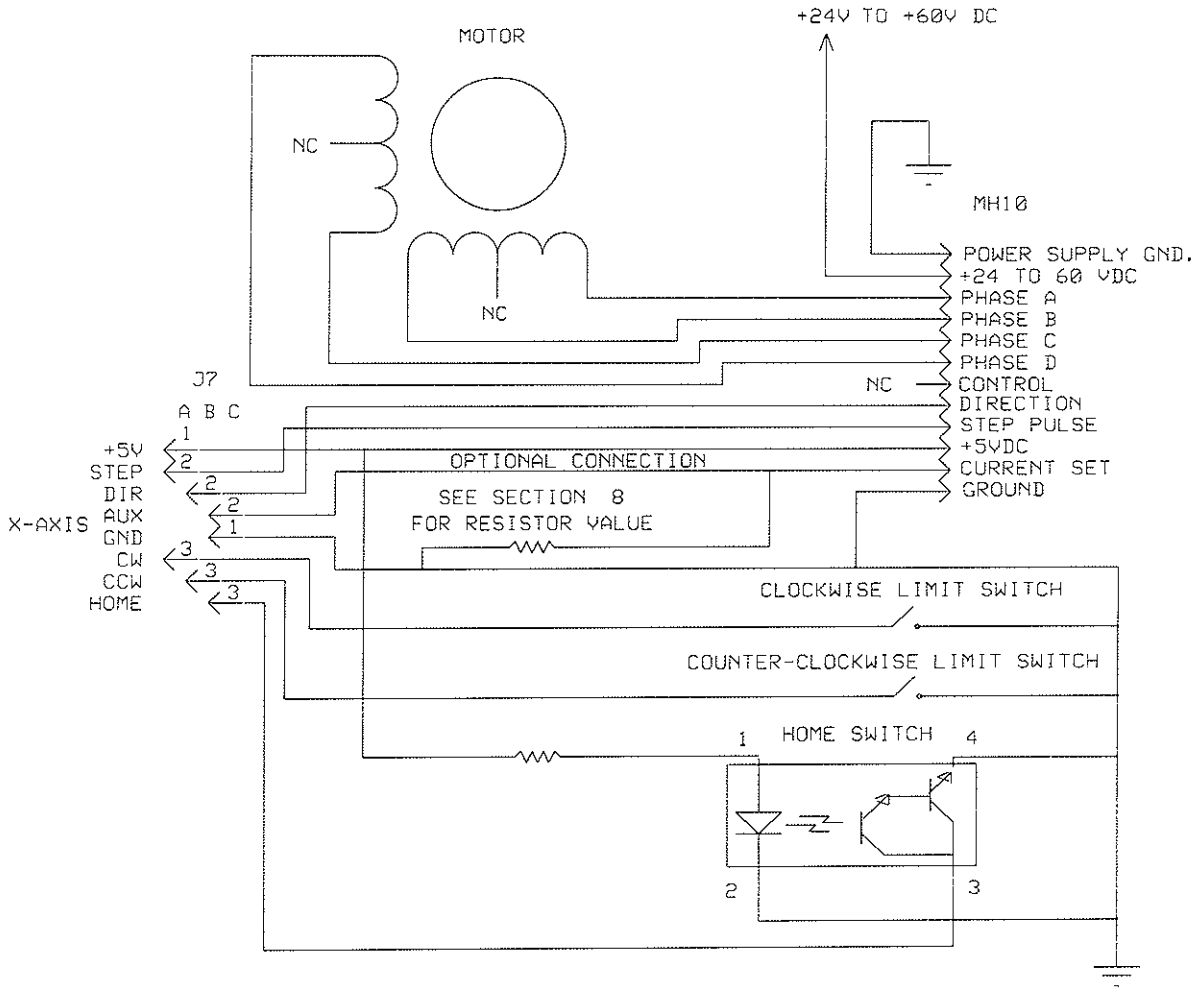


Figure 5-1 SRX/MH10 INTERCONNECTIONS

cause the motor to stop when the switch closes. On finding the home position the internal position counters will be initialized. The sense of the home switches may be changed to true when open, if desired, by use of the HH command. The limit switches may be changed to true when open, if desired, by removing the jumper on J28. Figure 5-1 shows a typical connection between an SRX board and a motor using an OMS MH10 motor driver.

6. ENCODER OPTION

6.1. INTRODUCTION

The encoder feedback option is intended primarily for applications where desired positional accuracy exceeds the accuracy of the mechanical drive components, such as lead screws, or when position feedback is required to detect motor slip or stall.

The encoder option accepts quadrature pulse outputs from high resolution encoders. Up to 50,000 pulse per revolution encoders may by used at frequencies up to 1 Mhz after quadrature detection. This allows position feedback information to match the resolution to the microstepping motor drive. The X and Y axes may be configured for encoder feedback on a two axis SRX equipped with option E or the X, Y, Z and T axes on four axis boards with option E.

6.2. MODES OF OPERATION

The SRX-E can monitor the actual position through the encoder pulse train. It can then correct for position errors due to system backlash or mechanical tolerances or report slip or stall of the motor to the host. A tracking mode is also provided which allows one axis to track the activity of another axis or positioning device. These options are selectable by the user through software commands.

6.3. ENCODER SELECTION AND COMPATIBILITY

The SRX with option E is compatible with virtually any incremental encoder which provides quadrature outputs. Times four quadrature detection is used to increase resolution. The inputs are compatible with encoders which have single ended TTL outputs as well as differential line drivers. Provisions are also provided for an index pulse (differential or single ended) and an index enable for systems requiring more than one revolution of travel (and thus multiple index pulses) from the encoder. A biasing network is provided on the board for termination of unused encoder inputs.

The user can specify the encoder count/motor count ratio for position maintenance and encoder tracking mode. This ratio is handled internally in floating point format and can be virtually any ratio. Slip detection requires that the encoder resolution (after the 4X quadrature detection) match the motor resolution.

6.4. ENCODER INTERFACE

The encoder connections are as shown in Tables 6-1 and 6-2.

Table 6-1 J7 CONNECTOR PIN ASSIGNMENT, ENCODER MODELS

PIN	ROW C FUNCTION	ROW B FUNCTION	ROW A FUNCTION
1	Ground	NC	+5VDC
2	X Auxiliary Output	X Direction Output	X Step Output
3	X Home Switch	X Positive Limit Switch	X Negative Limit Switch
4	Y Auxiliary Output	Y Direction Output	Y Step Output
5	Y Home Switch	Y Positive Limit Switch	Y Negative Limit Switch
6	Ground	NC	+5VDC
7	Z Auxiliary Output	Z Direction Output	Z Step Output
8	Z Home Switch	Z Positive Limit Switch	Z Negative Limit Switch
9	T Auxiliary Output	T Direction Output	T Step Output
10	T Home Switch	T Positive Limit Switch	T Negative Limit Switch
11	NC	NC	NC
12	Ground	NC	+5VDC
13	X Phase A-	X Phase A+	X Index +
14	X Phase B-	X Phase B+	X Index -
15	Y Phase A-	Y Phase A+	Y Index+
16	Y Phase B-	Y Phase B+	Y Index -
17	Ground	NC	+5VDC
18	Z Phase A-	Z Phase A+	Z Index +
19	Z Phase B-	Z Phase B+	Z Index -
20	T Phase A-	T Phase A+	T Index +
21	T Phase B-	T Phase B+	T Index _
22	NC	NC	NC
23	NC	NC	NC
24	Ground	Reset Input	+5VDC
25	User I/O Bit 16	User I/O Bit 8	User I/O Bit 0*
26	User I/O Bit 17	User I/O Bit 9	User I/O Bit 1*
27	User I/O Bit 18	User I/O Bit 10	User I/O Bit 2*
28	User I/O Bit 19	User I/O Bit 11	User I/O Bit 3*
29	User I/O Bit 20	User I/O Bit 12	User I/O Bit 4*
30	User I/O Bit 21	User I/O Bit 13	User I/O Bit 5*
31	User I/O Bit 22	User I/O Bit 14	User I/O Bit 6*
32	User I/O Bit 23*	User I/O Bit 15	User I/O Bit 7*

* = PLC Control Port

If single ended encoders are used, the unused line receiver inputs must be biased in the middle of the voltage swing of the active output. J17 is provided with a built in bias supply. The appropriate unused inputs should be connected to the +1.5VDC supply as needed. See Section 2 for more details on J17.

Please note that the U and V axes limit select jumpers (J28) and done flag register bit assignments on the -6E version controllers are as defined in the manual for the R and S axes.

Table 6-2 SRX-6E PIN ASSIGNMENTS

PIN	ROW C FUNCTION	ROW B FUNCTION	ROW A FUNCTION
1	Ground	NC	+5VDC
2	X Auxiliary Output	X Direction Output	X Step Output
3	X Home Switch	X Positive Limit Switch	X Negative Limit Switch
4	Y Auxiliary Output	Y Direction Output	Y Step Output
5	Y Home Switch	Y Positive Limit Switch	Y Negative Limit Switch
6	Ground	NC	+5VDC
7	Z Auxiliary Output	Z Direction Output	Z Step Output
8	Z Home Switch	Z Positive Limit Switch	Z Negative Limit Switch
9	T Auxiliary Output	T Direction Output	T Step Output
10	T Home Switch	T Positive Limit Switch	T Negative Limit Switch
11	NC	NC	NC
12	Ground	NC	+5VDC
13	X Phase A-	X Phase A+	X Index +
14	X Phase B-	X Phase B+	X Index -
15	Y Phase A-	Y Phase A+	Y Index+
16	Y Phase B-	Y Phase B+	Y Index -
17	Ground	NC	+5VDC
18	U Auxiliary	U Direction Output	U Step Output
19	U Home Switch	U Positive Limit Switch	U Negative Limit Switch
20	V Auxiliary Output	V Direction Output	V Step Output
21	V Home Switch	V Positive Limit Switch	V Negative Limit Switch
22	NC	NC	NC
23	NC	NC	NC
24	Ground	NC	+5VDC
25	User I/O Bit 16	User I/O Bit 8	User I/O Bit 0*
26	User I/O Bit 17	User I/O Bit 9	User I/O Bit 1*
27	User I/O Bit 18	User I/O Bit 10	User I/O Bit 2*
28	User I/O Bit 19	User I/O Bit 11	User I/O Bit 3*
29	User I/O Bit 20	User I/O Bit 12	User I/O Bit 4*
30	User I/O Bit 21	User I/O Bit 13	User I/O Bit 5*
31	User I/O Bit 22	User I/O Bit 14	User I/O Bit 6*
32	User I/O Bit 23*	User I/O Bit 15	User I/O Bit 7*

* = PLC Control Port

6.5. HOME PROCEDURES

Two logical inputs are provided to synchronize the physical hardware with the SRX controller, i.e. put the controlled motor in the home position.

The SRX home inputs can be used with encoders which provide one home pulse for the complete travel of the stage. This signal can be either a logic high or logic low true by using the SRX HH and HL commands. The HM or HR commands are used after reducing

the velocity to no more than 1024 pulses per second. This limit on velocity is necessary to avoid ambiguity of the home position if more than one pulse occurs per sample interval.

The index input on J7 uses internal logic to establish the home position when used with the HE command mode. This position consists of the logical AND of the encoder index pulse, the home enable external input (low true only) and a single quadrant from the encoder logic. The home enable pulse must be true for less than one revolution of the encoder, thus allowing only one home for the complete travel of the stage. The home logic expressed in boolean terms is:

$$home = phase\_A * /phase\_B * index * /home\_switch$$

Note that it is necessary that the above quadrant occur within the index pulse as provided by the encoder for this logic to function properly. It may be necessary with some encoders to shift the phase of this quadrant by inverting one or both of the phases. Inverting one phase or swapping phase A for phase B will also reverse the direction. The encoder counter (read by an RE command) must increase for positive moves or the system will oscillate due to positive feedback.

7. COMMAND STRUCTURE

7.1. INTRODUCTION

An extensive command structure is built into the SRX family of intelligent motor controls. It includes a 200 command and parameter buffer for each axis and a command loop counter which allows multiple executions of any command string.

The following commands in this section are included in the SRX family of controllers. All the commands are two ASCII characters and may be in upper or lower case. Some of the commands expect a numerical operand to follow. These commands are identified with a '#' after the command. The operand must be terminated by a space, carriage return or semi-colon to indicate the end of the number. No terminator is required on the other commands, but may be included to improve readability. The operand must immediately follow the command with no space or separation character. The '#' indicates a signed integer input parameter or a signed fixed point number of the format ##.# when user units are enabled. With user units enabled distances, velocity and acceleration parameters may be input in inches, revolutions, etc.

Synchronized moves may be made by entering the AA command. This command performs a context switch which allows entering the commands in the format MRx#,y#,z#,t#,u#,v#,r#,s#;. Numbers are entered for each axis which is to be commanded to move. An axis may be skipped by entering the comma with no parameter. The command may be prematurely terminated with a ";", i.e. a move requiring only the X and Y axes would use the command MRx#,y#; followed by the GO command. Each axis programmed to move will start together upon executing the GO command. The SRX can be switched back to the unsynchronized mode by entering the desired single axis command such as AX.

The AM command is provided for complex applications where the host manages multiple motion processes by a multitasking operating system. This mode shares the same instructions as the AA mode, but allows starting a task while some other task involving one or many axes is active. For example, the X and Y axes could be doing linear interpolation while the Z axis is making an unrelated move simultaneously.

Constant velocity contouring provides another mode wherein the move parameters are predefined by entering AA then CD#,#;. The SRX will then calculate the move profile in advance and move at constant velocity in the prescribed pattern. It can do linear interpolation on as many as 8 axes between the predefined points or it can do circular interpolation mixed with linear on two axes.

7.2. COMMAND QUEUES

The input characters are placed in a character buffer on input then removed and interpreted. The commands are then placed in separate circular queues for each axis. As they are executed the space is reclaimed allowing the host to pass commands ahead of the

moves actually being processed. Most of the commands are placed in the appropriate command queue for execution while others are executed immediately allowing return of status information in a timely way rather than when encountered in the command stream. This information is provided in a table for each command which shows the queue requirements if any and indicates immediate in those cases where the command is not queued. The single axis cases are indicated by the mode reference indicating the appropriate axis. The synchronized mode is indicated by the mode identifier AA or AM. The contouring case is indicated by AA/CD for multiple axes in contour definition mode. The RQ command may be used to determine the actual space available at any time. The queues operate independently allowing each axis to perform separate processes simultaneously. The synchronized modes (AA) insert special wait opcodes which allow the axes to be synchronized in this mode. When the commands are nested within loops, the queue space is not reclaimed until after the loop has been executed the programmed number of times. For loops larger than the queue space, the loop may never be completed since it cannot reclaim the the queue space and cannot accept the loop terminator. The RQ command may be used to examine the remaining queue space. A Control-D may clear this condition if the input character queue is not also filled since it bypasses the command interrupter.

The following commands are available in firmware revision 1.75 and above.

Besides the responses of the SRX board to the commands listed here, the SRX will also send certain characters to inform the host when an event occurs. They are as follows:

FLAG	DESCRIPTION
!	Signals an execution of an ID command signalling that a move is complete.
@	Signals an overtravel condition showing that a limit switch has been triggered.
#	Signals a command error when echo is turned off.
$	Signals that the encoder has slipped and no longer matches the position counter.

7.3. AXIS SPECIFICATION COMMANDS

The following commands set the context to direct the commands which follow to the appropriate axis. They remain in effect until superseded by another command of the same type, specifying a different axis.

AA AXES ALL

The AA command will perform a context switch to coordinated moves.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	2
AA/CD	Not valid

Example: Perform an absolute move using the X and Y axes.

Enter: AA MR12000,14000; GO

AM AXES MULTITASKING

The AM mode allows several tasks to be managed simultaneously. For instance, a task may be performing coordinated motion on 2 axes, while a second task is performing unrelated but simultaneous motion on another axis.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Perform a coordinated move on the X and Y axes, while moving the T axis as a separate move.

Enter: AM MR2000,3000; GO MA,,,10000; GO

AX AXIS X

The AX command sets the context to direct all the following commands to the X axis. This is the default mode at power up or reset.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Make the X axis step at a rate of 5,000 steps/second.

Enter: AX JG5000;

AY AXIS Y

The AY command sets the context to direct all the following commands to the Y axis.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Examine the status of the Y axis.

Enter: AY RA

AZ AXIS Z

The AZ command sets the context to direct all the following commands to the Z axis.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Move the Z axis 2,000 steps at a rate of 500 steps/second.

Enter: AZ VL500 MR2000 GO

AT AXIS T

The AT command sets the context to direct all the following commands to the T axis.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Move the T axis to absolute position -2468.

Enter: AT MA-2468; GO

AU AXIS U

The AU command sets the context to direct all the following commands to the U axis.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Set the U axis position register to -56789.

Enter: AU LP-56789

AV AXIS V

The AV command sets the context to direct all the following commands to the V axis.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Set the auxiliary line low on the V axis.

Enter: AV AF

AR AXIS R

The AR command sets the context to direct all the following commands to the R axis.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Examine the queue size of the R axis.

Enter: AR RQ

AS AXIS S

The AS command sets the context to direct all the following commands to the S axis.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Stop all movement on the S axis only.

Enter: AS ST

7.4. SYSTEM CONTROL COMMANDS

These commands allow control of various system parameters and operating modes to allow the user to optimize the response of the system for his/her application needs.

EN ECHO ON

The EN command enables echoing. All commands and parameters will be echoed to the host. This mode is useful for debugging command strings from a terminal. This mode also outputs an English readable error message to the host which may be echoed to the terminal or computer to aid in debugging.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Not valid
AA/CD	Not valid

Example: Enable echoing by the SRX so that commands are echoed and the error message are returned to the host as a readable ASCII string. This command would probably be the first command executed after turning on the system when this mode is desired.

Enter: EN

EF ECHO OFF

The EF command disables echoing from the SRX motion system. This is the default mode at power up or reset.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Not valid
AA/CD	Not valid

Example: Stop echoing to the host.

Enter: EF

HH HOME HIGH

The HH command sets the sense of the home switch on the current axis to active high. This allows the use of a normally closed switch.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	Not valid
AA/CD	Not valid

Example: (see HL the following command)

HL HOME LOW

The HL command sets the sense of the home switch on the current axis to active low. This is the default mode at power up or reset.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	Not valid
AA/CD	Not valid

Example: A faster home sequence may be used in applications which have a long distance to travel to reach home. The stage is moved through home at high speed with the home switch set for active high then reversed at low speed to meet the 1024 steps per second requirement of the home command.

Enter:
```
AX VL20000 HH HM0
VL1000 HL HR0
```

LF LIMITS OFF

The LF command turns off the limit switches for the addressed axis. This allows the stage to move beyond the limit switch and should be used with caution.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	Not valid
AA/CD	Not valid

Example: Set up a board to ignore the Y axis limit switches.

Enter: AY LF

LN LIMITS ON

The LN command restores the operation of the limit switches for the addressed axis. This is the default mode at power up or reset.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	Not valid
AA/CD	Not valid

Example: Set up the T axis to stop immediately when a limit switch is encountered.

Enter: AT LN

SL SOFT LIMIT

The SL command changes the operation of the limit inputs causing the output pulse train to ramp down instead of terminating immediately. The output queue is not flushed except for the current move. This mode is effective for point to point moves only. This command is valid in the single axis mode only, but affects all axes simultaneously.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	Not valid
AA/CD	Not valid

Example: Set up a board to allow each axis to ramp to a stop when a limit is encountered.

Enter: AX SL

SF SOFT LIMIT OFF

The SF command restores the normal operation of the limit switches. This is the default mode at power up or reset.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	Not valid
AA/CD	Not valid

Example: Set up a board to make each axis stop immediately when a limit is encountered.

Enter: AX SF

CN COSINE ON

The CN command enables cosine velocity ramps, i.e. half sinusoid acceleration profiles for all axes. The cosine is not truncated in moves that do not reach full speed. See Section 1 for explanation of velocity profiles. This command should not be given while an axis is in motion or the results may not be predictable. This command affects all axes, even if issued in the single axis mode.

Because of the excess processing overhead involved, absolute moves, such as MA and MT, cannot be used within loops (LS-LE, WH-WG) while the board is in the cosine (CN) velocity profile mode. Relative moves, such as MR and ML, will work properly within loops, when in the cosine mode.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

PN# PARABOLIC ON

The PN command sets all axes to truncated parabolic ramps. This acceleration profile starts at 100% of the programmed acceleration and decreases in steps of 10% of the initial acceleration down to as low as 10%. The parameter supplied selects the number of steps. It must be in the range of 3 to 10 corresponding to 70% and 10% acceleration at the peak respectively. A parameter out of this range or no parameter supplied defaults to 70% or 3 steps. Note that the parameter is the number of steps, not the acceleration values. The larger number is a lower acceleration at the peak. See Section 1 for explanation of velocity profiles. This command should not be given while an axis is in motion or the results may not be predictable. This command affects all axes, even if issued in the single axis mode.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Set the board to be in the smoothest parabolic acceleration ramp.

Enter: PN10;

PF PARABOLIC OFF

The PF command restores all axes to linear acceleration and deceleration ramps. This is the default mode at power up or reset. See Section 1 for explanation of velocity profiles. This command should not be given while an axis is in motion or the results may not be predictable. This command turns off the PN and CN modes. This command affects all axes, even if issued in the single axis mode. This is the default mode at power up or reset.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Turn off cosine or parabolic ramps, returning to linear.

Enter: PF

RR RESET RAM

The RR command resets all variables to their defaults, clears all macros and position memory cells, then performs a hardware reset like the RS command.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Clear all macros.

Enter: RR

RS RESET

The RS command is a software reset which causes the local SRX microprocessor to reset. All previously entered data and commands are lost. All internal parameters are initialized to defaults, unless non-volatile RAM is installed. This command is intended for catastrophic failure recovery only. The KL command should be used to reset queues or return the system to a known state.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Clear everything in the board and stop all movement. Reset all hardware registers.

Enter: RS

SB# SET BAUD RATE

The SB command allows the user to change the baud rate. The parameter is the desired baud rate in bits per second. The rate is changed immediately. The default is 9600 bits per second. The rate is stored in non-volatile RAM and is only reset to the default by an RR command. Valid baud rates are 300, 600, 1200, 2400, 4800, 9600, and 19200.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Not valid
AA/CD	Not valid

Example: Set baud rate to 19200.

Enter: SB19200;

IT INITIALIZE DEFAULTS

The SRX does not initialize to the default velocity and acceleration parameters on power up or reset as the other OMS controls. This is undesirable in the stand alone mode since there is no way to initialize these parameters without a host computer or terminal. The IT command resets the velocity to 200,000 steps per second and the acceleration to 2,000,000 steps per second per second.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Not valid
AA/CD	Not valid

Example: Set velocities and accelerations to default values.

Enter: IT

7.5. USER I/O COMMANDS

The following commands are for accessing the bit I/O functions of the board. See also the SW and WS commands.

AN AUXILIARY ON

The AN command turns on the selected auxiliary output ports. That is, it allows the open collector line to be pulled high by an external pull up resistor. The AN command may be used to change power level on driver modules so equipped, trigger another board's input or as a user specified output. This is the default mode for the auxiliary line at power up or reset.

A parameter must be supplied for the desired axes when used in the AA mode so that the other axes are not affected. The parameter only serves as a place holder to show which axes should be affected, the value given does not affect the active state of the auxiliary line. No parameter is required in the single axis mode.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	1
AA/CD	2

Example: Turn on the Y axis auxiliary output in the single axis mode.

Enter AY AN

Example: Turn on the X and Z axes auxiliary outputs when in the AA command mode. The Y axis is unchanged in this example.

Enter: AA AN1,,1;

AF AUXILIARY OFF

The AF command turns off the selected auxiliary outputs. That is, it causes the open collector line to be driven low. The AF command may be used to change power level on driver modules so equipped or as a user specified output. Same parameter rules apply as the AN command.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	1
AA/CD	2

Example: Turn off the Y axis auxiliary output in the single axis mode.

Enter: AY AF

Example: Turn off the X and Z axes auxiliary outputs when in the AA command mode. The Y axis is unchanged in this example.

Enter: A AF1,,1;

PA# POWER AUTOMATIC

The PA command will turn on or off the auxiliary outputs at the beginning of each GO or GD command execution and complement the outputs after the move is executed. The auxiliary will be turned on, i.e. pulled high, upon the execution of the GO or GD and off at the end of that move, if the parameter is zero or not specified in the single axis mode. If the parameter is non-zero, the sense is reversed, i.e. the auxiliary output is turned off (driven low) upon the execution of the GO or GD command and on at the end of the move.

This mode need only be set once and can be turned off by using the AN or AF command. Axes can be selectively affected in the AA mode by following the syntax as described for the AN command. The values of the included parameters set the state of the auxiliary line during the move. The following queue requirements apply to each GO or GD command in the command stream in the AA and single axis modes.

QUEUE REQUIREMENTS	
MODE	
AX - AS	2
AA,AM	2
AA/CD	Not valid

Example: Turn on the Y axis auxiliary output at the beginning of a move and turn the T axis output off at the beginning of a move, while in the AA command mode.

Enter: AA PA,0,,1;

SE# SETTLING TIME

The SE command allows specification of a settling time, in milliseconds, to be used before the power is reduced, when using the PA mode. The parameter may be any value to 1000 milliseconds. Specification of a parameter of zero turns off the mode. This command is available in single axis mode only. The use of this command requires 3 queue slots with the execution of each GO or GD command.

QUEUE REQUIREMENTS	
MODE	
AX - AS	3
AA,AM	3
AA/CD	Not valid

Example: Turn on the Z axis auxiliary output upon execution of a move and have it remain on for 500 milliseconds after the move is complete.

Enter: AZ PA SE500;

BL# BIT LOW

The BL command sets the selected user output on (i.e. logic low).

QUEUE REQUIREMENTS	
MODE	
AX - AS	2
AA,AM	2
AA/CD	Not valid

Example: Turn on output bits 16 and 18 after a move. Note that this is only valid for bits which have been configured as outputs. See the RB command in this section.

Enter: AX MA1000 GO BL16; BL18;

BH# BIT HIGH

The BH command sets the selected general purpose output off (i.e. logic high). The state of general purpose outputs is off at power up or reset. Valid bits depend on which bits are programmed as outputs. Factory default output bits are 16 through 23.

QUEUE REQUIREMENTS	
MODE	
AX - AS	2
AA,AM	2
AA/CD	Not valid

Example: Set general purpose output bits 17 and 22 high.

Enter: BH17; BH22

BX BIT REQUEST IN HEX

The BX command returns the state of the general purpose I/O bits, in a six digit hex format, surrounded by line feed and carriage return pairs. A one in any binary position signals that bit as being low.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: BX

Response: <LF><CR>810001<LF><CR>
This shows bits 23, 16, and 0 are low, the rest are high.

RB REQUEST BIT DIRECTION

The RB command returns the direction of the I/O bits as they are currently configured on the board in hex format. Output bits return a 1 while input bits return a 0. The six hexidecimal digit value is surrounded by line feed and carriage return pairs. Each character returned represents four I/O bits. A response of FF0000 would indicate that bits 0 through 15 are inputs and I/O bits 16 through 23 are outputs, which is the factory default.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	Not valid
AA/CD	Not valid

Example: Inquire the board for the directions of the I/O bits.

Enter: RB

Response: <LF><CR>FF0000<LF><CR>

7.6. MOVE SPECIFICATION COMMANDS

These commands allow specification of move parameters. They allow move parameters to be tailored to the user's system requirements.

AC# ACCELERATION

The AC command sets the acceleration/deceleration register to the operand which follows the command. The parameter must be greater than zero and less than 8,000,000. All the following move commands for the axis being programmed will accelerate or decelerate at this rate until another AC command is entered. All acceleration registers default to 2,000,000 steps per second per second upon power-up or reset, unless non-volatile RAM is installed.

The acceleration register may be automatically modified by the SRX if an ML or MT instruction is sent in the AA or AM modes. The user must then redefine them with an AC command, when returning to the single axis mode, or when using move commands in the AA or AM modes which do not do interpolation, such as the MA or MR commands.

QUEUE REQUIREMENTS			
MODE	**LINEAR**	**PARABOLIC**	**COSINE**
AX - AS	4	15	15
AA,AM	4	15	15
AA/CD	Not valid		

Example: In the single axis mode, set the Y axis acceleration to 200,000 counts per second per second.

Enter: AY AC200000

Example: In the AA mode, set the acceleration of the X axis to 200,000 and the Z axis to 50,000 and leave the other axes with their previous values.

Enter: AA AC200000,,50000;

VL# VELOCITY

The VL command sets the maximum velocity register of the axis being programmed to the operand which follows the command. The operand must be greater than zero and less than or equal to 522,000 steps per second. The velocity defaults to 200,000 at power up or reset, unless non-volatile RAM is installed. This is a write only register and controls the maximum velocity used in relative and absolute position moves except as modified by the linear interpolation instructions.

If the velocity register is modified by an ML or MT instruction in the AA or AM modes, the user must redefine the velocity with a VL command when returning to the single axis mode or using a move command which does not use interpolation in the AA or AM modes.

QUEUE REQUIREMENTS			
MODE	**LINEAR**	**PARABOLIC**	**COSINE**
AX - AS	2	13	13
AA,AM	2	13	13
AA/CD	Not valid		

Example: In the single axis mode, set the X axis velocity to 10,000 counts per second per second.

Enter: AX VL10000

Example: In the AA mode, set the peak velocity of the X axis to 5,000 and the T axis to 50,000 and leave the other axes with their previous values.

Enter: AA VL5000,,,50000;

VB# VELOCITY BASE

The VB command allows the velocity ramp to start at the specified velocity. This allows faster acceleration and the ability to pass through resonance quickly in some applications. The velocity jumps instantly to the specified velocity, then ramps as usual. The deceleration is the same in reverse. This mode is active only for linear ramps. It is ignored for cosine and parabolic ramps but not flagged as a command error. The parameter must be greater than zero and less than the programmed velocity. This command is not valid with the JG command. The base velocity defaults to zero at power up or reset, unless non-volatile RAM is installed.

QUEUE REQUIREMENTS			
MODE	**LINEAR**	**PARABOLIC**	**COSINE**
AX - AS	2	2	2
AA,AM	2	2	2
AA/CD	Not valid		

Example: In the single axis mode, set the Y axis velocity base to 200.

Enter: AY VB200

Example: In the AA mode, set the X and Y axes velocity bases to 200.

Enter: AA VB200,200;

LP# LOAD POSITION

The LP command will immediately load the position supplied as a parameter in the absolute position register of the axis. In models with the encoder option, the parameter will be loaded into the encoder position register and the parameter times the encoder ratio will be loaded into the position counter. If the no parameter is supplied, the value of zero is used. This command turns off the position hold and interrupt on slip modes when used in an SRX with the encoder option.

QUEUE REQUIREMENTS		
MODE	**NO ENCODER**	**ENCODER**
AX - AS	2	4
AA,AM	Not valid	
AA/CD	Not valid	

Example: The following would load the X axis position register with 1000.

Enter: AX LP1000

Example: The following would load the Y axis position register with 20,000 and the encoder position register with 30,000 counts, in encoder models.

Enter: AY ER3,2 LP30000

MA# MOVE ABSOLUTE

The MA command will set up the axis to move to the absolute position supplied as a parameter. The default value of zero is used if no parameter is supplied in the single axis mode. In the AA mode, an axis may remain stationary by entering a comma but omitting the parameter. The move is actually initiated by a GO or GD command.

In the AA mode, each axis will use its predefined acceleration and velocity values to move to the new absolute position. Each axis may, or may not, get to the destination at the same time, because each axis utilizes individual velocities and accelerations.

Because of the excess processing overhead involved, the MA command cannot be used within loops (LS-LE, WH-WG) while the board is in the cosine (CN) velocity profile mode.

The linear move commands (ML and MT) and the constant velocity mode may alter predefined acceleration and velocity values. These values should be redefined if you go from a linear move to a non-linear move, such as an MA or MR type, in both single axis or all axes modes.

QUEUE REQUIREMENTS			
MODE	**LINEAR**	**PARABOLIC**	**COSINE**
AX - AS	2	2	2
AA,AM	2	2	2
AA/CD	Not valid		

Example: Move the X axis to absolute position 100,000 counts with the previously entered acceleration and velocity parameters.

Enter: AX MA100000 GO

Example: In the AA mode, move the Y axis to absolute position 10,000 counts and the T axis to absolute position 1,000 counts. The other axes will remain in their current positions.

Enter: AA MA,10000,,1000; GO

MR# MOVE RELATIVE

The MR command will set up the axis to move relative from the current position at the time the move is executed. In the AA mode, an axis may remain stationary by entering a comma but omitting the parameter. The move is actually initiated by a GO or GD command.

In the AA mode, each axis will use its predefined acceleration and velocity values to move to the new absolute position. Each axis may, or may not, get to the destination at the same time, because each axis utilizes individual velocities and accelerations.

The linear move commands (ML and MT) and the constant velocity mode may alter predefined acceleration and velocity values. These values should be redefined if you go from a linear move to a non-linear move, such as an MA or MR type, in both single axis or all axes modes.

QUEUE REQUIREMENTS			
MODE	**LINEAR**	**PARABOLIC**	**COSINE**
AX - AS	2	2	2
AA,AM	2	2	2
AA/CD	Not valid		

Example: In the single axis mode, move the X axis 2468 steps in the negative direction.

Enter: AX MR-2468 GO

Example: In the AA mode, move the X axis 12345 steps in the positive direction and the Y axis 6789 steps in the positive direction. Both axes will start at the same time.

Enter: AA MR12345,6789; GO

ML#,#; MOVE LINEAR

The ML command uses linear interpolation to perform a straight line relative move to the new location. Input parameters are relative distance for each axis in the move. Velocity and acceleration parameters of each axis may be automatically adjusted by the SRX controller to perform the linear move. If linear and single axis moves are mixed, it will be necessary to reset the velocity and acceleration parameters for the single axis move following a linear move.

The parameters may have been modified by the SRX depending on the relative distances of the linear move. The ML command should be followed by a GO or GD to start the axes together. The velocity and acceleration parameters are scaled to allow the axes to move and finish together. All axes are scaled to the axis with the longest move time.

QUEUE REQUIREMENTS			
MODE	**LINEAR**	**PARABOLIC**	**COSINE**
AX - AS	Not valid		
AA,AM	6	30	30
AA/CD	Not valid		

Example: In the AA mode, move the Y, Z and T axes 10000, 100 and 1000 counts respectively with each starting and finishing together. The other axes remain in their previous positions.

Enter: AA ML,10000,100,1000; GO

MT#,#; MOVE TO

The MT command uses linear interpolation to move to the specified absolute position. The syntax is similar to the ML command. This command is invalid while in the CN mode, if loops are being used. The command will become valid again after executing an ST or KL command. The MT command is not valid in loops (LS-LE, WH-WG) at anytime. When used in the contour definition mode, only the axes being used in the contour must be provided for in the MT syntax. A GO or GD command initiates the move.

The MT command may alter predefined acceleration and velocity values. These values should be redefined if you go from a linear move to a non-linear move, such as an MA or MR type, in both single axis or all axes modes.

QUEUE REQUIREMENTS			
MODE	**LINEAR**	**PARABOLIC**	**COSINE**
AX - AS	Not valid		
AA,AM	6	30	30
AA/CD	4 + number of axes		

Example: In the AA mode, move the X, Y and T axes to absolute positions 1000, 10000 and 100 counts respectively, with each starting and finishing together. The unused axes remain in their previous positions.

Enter: AA MT1000,10000,,100; GO

MO MOVE ONE PULSE

The MO command will output one step pulse in the current direction (do not use the GO command). The direction may be reversed by use of the MM or MP command. This command generates the output pulse in one sample interval and thus eliminates the latency of generating a ramp with an MR1 GO command sequence. This command is not available in models with an encoder option.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	Not valid
AA/CD	Not valid

Example: Move the Z axis one pulse in the negative direction.

Enter: AZ MM MO

RM# REMAINDER

The RM command will divide the position counter by the parameter supplied and replace the position counter with the resulting remainder. The parameter must be greater than zero and less than 65000. This command is used in applications where the controller is managing the motion of a continuously rotating object. It allows the position counter to keep track of the absolute position without regard to the number of revolutions it may have rotated. This command has no effect on the encoder position register on boards with the encoder feedback option.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	Not valid
AA/CD	Not valid

Example: An RM2000 command with a position counter of -4050 will return a position of 1950 since it is within 50 counts of rolling over at -4000, i.e. the axis is 1950 counts from the starting point.

7.7. MOVE EXECUTION COMMANDS

These commands allow execution of the moves which have been previously specified.

GO GO

The GO command will initiate the move which has been previously programmed with such commands as MA, MR, MT, and ML. No operand is required with the GO command.

To find the total queue requirements for a specific application, find the appropriate value in Table A. If the board is an encoder version, add the value found in Table B to the value from Table A, to determine total queue usage.

TABLE A		
QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AS	4	7
AA,AM	5	8
AA/CD	Not valid	

TABLE B			
ADDITIONAL ENCODER QUEUE REQUIREMENTS			
MODE	LINEAR	PARABOLIC	COSINE
AX - AS	0	0	0
AA,AM	6	15	15
AA/CD	Not valid		

Example: In the single axis mode, move the X axis to absolute position 12345.

Enter: AX MA12345 GO

Example: In the AA mode, move the X axis 2468 steps in the positive direction and the Y axis 2468 steps in the negative direction.

Enter: AA MR2468,-2468; GO

GD GO and RESET DONE

The GD command may be substituted for a GO command. It will reset the done flags, then initiate the move which has been prevously programmed with such commands as MA, MR, MT, and ML; just as the GO command does. In the single axis mode, only the done flag for the selected axis will be reset.

In the AA mode, all the done flags will be reset. In the AM mode, the axes involved in the move will be reset. This allows the host to reset the interrupts on the axis involved in the next move, without affecting other axes which may be still active. Note that this command is probably only useful in applications where commands are queued in advance, since the interrupt may be reset before the host has the opportunity to service it, if the GD command is waiting in the queue.

To find the total queue requirements for a specific application, find the appropriate value in Table A. If the board is an encoder version, add the value found in Table B to the value from Table A, to determine total queue usage.

QUEUE REQUIREMENTS		
MODE	**NO ENCODER**	**ENCODER**
AX - AS	5	8
AA,AM	6	9
AA/CD	Not valid	

TABLE B			
ADDITIONAL ENCODER QUEUE REQUIREMENTS			
MODE	**LINEAR**	**PARABOLIC**	**COSINE**
AX - AS	0	0	0
AA,AM	6	15	15
AA/CD	Not valid		

Example: In the single axis mode, move the Y axis 12345 steps in the negative direction and set the done flag when the move is completed. Then move it 12345 steps in the positive direction, clear the previous done flag and set the done flag, again, when the move is completed.

Enter: AY MR-12345 GO ID MR12345 GD ID

Example: In the AA mode, perform a linear absolute move with the X and Y axes to the position 10000,20000 and set the done flag when the move is completed. Then perform a linear relative move on both axes, moving the X axis 10000 steps in the negative direction and the Y axis 20000 steps in the negative direction.

Enter: AA MT10000,20000; GO ID ML-10000,-20000; GD ID

JG# JOG

The JG command is a velocity mode and will step the axis at the velocity supplied as a parameter. The JG command will accelerate to the programmed velocity and run until altered by an ST, SA, KL or another JG command. The jog velocity may be changed by following the command with another JG command of a different velocity. The axis must be stopped before reversing directions. This command modifies the move velocity parameter (VL) for the affected axis. The JG command does not require a GO or GD command to start the motion.

QUEUE REQUIREMENTS			
MODE	**LINEAR**	**PARABOLIC**	**COSINE**
AX - AS	2	Linear ramp	
AA,AM	Not valid		
AA/CD	Not valid		

Example: Jog the motor at 100,000 steps per second then change to 35,000 steps per second when the second JG is entered, then stop by decelerating to a stop.

Enter: JG100000 JG35000 ST

JF# JOG FRACTIONAL VELOCITIES

The JF command will jog the axis at the velocity specified, like the JG command. The parameter may include a fractional part allowing better resolution at low speeds. The velocity set by this command will remain the default velocity until altered by a VL, JG or another JF command.

QUEUE REQUIREMENTS		
MODE	**NO ENCODER**	**ENCODER**
AX - AS	2	3
AA,AM	Not valid	
AA/CD	Not valid	

Example: Jog the Y axis at 2⅔ steps per second.

Enter: AY JF2.667

VS#,#,# VELOCITY STREAMING

The VS command will generate a pulse train without acceleration or deceleration at the rates specified. The parameters are time in 1/1024 second sample intervals, X velocity, and Y velocity. This is a slave mode and cannot be mixed or queued with other commands. You must be in the AX mode, since the VS command and all parameters are inserted in the X axis command queue. The VS command does not require a GO command to start the motion.

QUEUE REQUIREMENTS	
MODE	
AX	5
AY - AS	Not valid
AA,AM	Not valid
AA/CD	Not valid

Example: Create a stair step ramp on the X and Y axes, with the X axis moving in the negative direction and the Y axis in the positive direction. Make each step last 1 second long and increase velocity by 1,000 steps/second, until a velocity of 3,000 steps/second is reached, then step down to 0 steps/second.

Enter: AX VS1024,-1000,1000; VS1024,-2000,2000; VS1024,-3000,3000; VS1024,-2000,2000; VS1024,-1000,1000; VS1,0,0;

7.8. MOVE TERMINATION COMMANDS

The following commands allow termination of move sequences in process.

ST STOP

The ST command flushes the queue for the current axis only, in the single axis mode, and causes the axis to decelerate to a stop at the rate previously specified in an AC command. This command is used to stop the motor in a controlled manner from the jog mode or an unfinished GO or GD command. This command is executed immediately. All status and position information is retained. When executed in the AA mode, the ST command is equivalent to the SA command.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Flush + 2
AA,AM	Flush + 2
AA/CD	Not valid

Example: Move the Y axis for a while at 1200 steps/second, then ramp to a stop.

Enter: AY JG1200 (wait awhile) ST

SA STOP ALL

The SA command flushes all queues and causes all axes to decelerate to a stop at the rate previously specified in an AC command. All status and position information is retained.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Flush + 2
AA,AM	Flush + 2
AA/CD	Not valid

Example: Send all axes on a move, then ramp them to a stop, before they finish.

Enter: AA VL100,100,100,100,100,100,100,100;
MR1000,2000,3000,4000, 5000,6000,7000,8000; GO (wait awhile)
SA

SD STOP AND RESET DONE

The SD command may be substituted for the SA command. It will reset the done flags, then proceed to stop all axes. This allows the host to be interrupted when all axes have stopped by using the ID command after the SD. The SA ID combination may flag the completion early if one of the axes is already done from a previously executed ID.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Flush +2
AA,AM	Flush +2
AA/CD	Not valid

Example: Flag a done when all axes have stopped.

Enter: AA SD ID

KL KILL

The KL command will flush the command queue and terminate pulse generation of all axes immediately. It is intended for emergency termination of any program and to reset the input queues to a known state. The motor may not stop immediately even though no more pulses are delivered due to inertia of the motor rotor and load. Therefore, the position counter may not accurately reflect the true position of the motor following this command. The homing sequence should be used to reestablish the position counters. A Control-D (ASCII 4) will perform the same functions as the KL command. It bypasses the command interpreter and may work when the character buffer is full and the KL command cannot get through the interpreter. A Control-D should be used instead of KL, when the board appears hung-up. This can occur when its input queue is inadvertently filled, by entering a loop sequence that was so long you could not enter the LE command.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Flush + 2
AA,AM	Flush + 2
AA/CD	Not valid

Example: Stop all previously defined movement and flush the queue of a partially entered incorrect move command (you wanted a negative move not a positive one), before GO is entered.

Enter: AX MR5000 (oops!) KL MR-5000 GO

7.9. LOOP CONTROL COMMANDS

These commands allow move sequences to be repeated within loops. Loops can be nested up to four levels deep on each axis.

LS# LOOP START

The LS command sets the loop counter for the axis being programmed in the single axis mode and all axes in the AA mode. The command expects a loop counter operand following the command. The commands up to the LE loop terminator will be executed the number of times specified by the operand. Loops may be nested up to four levels deep on each axis. The parameter must be less than 32,000.

The first loop of commands will occur immediately as they are entered. The remaining loops will be executed after the loop terminator LE has been entered.

Because of the excess processing overhead involved, the MA command cannot be used in the loop mode, while the board is in the cosine (CN) velocity profile mode, and the MT command cannot be used in the loop mode at any time.

The axis mode (e.g. AX, AY, AA) must be the same when entering and exiting the loop, otherwise the matching loop termination command will not be found by the board's command processor.

If you want one axis to wait for another in the loop, you must be in the AA mode throughout the loop. If you are in the single axis mode in the loop, each axis' commands will go into their separate queues and execute independently of each other.

Another important thing to note is that the command queue size is 200. Each queued command takes one or more slots. If, when entering a looping sequence of commands, all 200 queue slots are filled, before the LE loop terminator is entered, the board will hang. This is because there is no space for the LE command, or any other commands. To clear this hang up, send the board a Control-D (same as KL, but shorter) to kill all moves and flush all queues. When programming a loop of more than four or five moves, the queue size should be examined with the RQ command to see if it is nearing zero.

QUEUE REQUIREMENTS	
MODE	
AX - AS	2
AA,AM	2
AA/CD	Not valid

Example: Execute a 100,000 count relative move on the Z axis 5 times.

Enter: AZ LS5 MR100000 GO LE

NOTE: The first move will occur immediately after entering the GO command. The remaining 4 moves will be executed after the loop terminator LE has been entered.

Example: Execute a 100,000 count move relative on the X axis together with a 100 count move on the T axis, followed by a move absolute to 100 counts on the X axis and 200 counts on the T axis, four times.

Enter: AA LS4 MR100000,,,100; GO MA100,,,200; GO LE

LE LOOP END

The LE command terminates the most recent LS command. The axis will loop back and repeat the commands within the loop the number of times specified in the LS command. The loop will start repeating as soon as this command is terminated.

QUEUE REQUIREMENTS	
MODE	
AX - AS	2
AA,AM	2
AA/CD	Not valid

Example: (see LS command on the previous page)

WS# WHILE SYNC

The WS command will execute the commands between the WS and WD commands as a loop while the specified general purpose input line is true, i.e. low. When the line goes high it will exit the loop and execute the commands which follow. The test is at the bottom of the loop, i.e. it will always be executed at least once. Valid I/O lines depend on which lines are configured as inputs. Factory defaults are 0 through 15.

QUEUE REQUIREMENTS	
MODE	
AX - AS	2
AA,AM	2
AA/CD	Not valid

Example: Execute a continuous loop, moving the X axis 10,000 counts and then move the Y axis -1000 counts, until an external device terminates the loop.

Enter: AA WS1 MR 10000; GO MR, -1000; GO WD

WD WHILE END

The WD command serves as the loop terminator for the WS command.

QUEUE REQUIREMENTS	
MODE	
AX - AS	2
AA,AM	2
AA/CD	Not valid

Example: (see WS command above)

WH WHILE

The WH command will execute all commands between it and the terminating WG command as a loop until terminated by a CW command. This allows repeated execution of a command sequence which can be terminated by the host. These commands may not be nested but may be executed sequentially.

QUEUE REQUIREMENTS	
MODE	
AX - AS	3
AA,AM	3
AA/CD	Not valid

Example: You have a 3 axis platform that you use to drill holes in the center of a 1/4 inch thick sheet of metal. The sheet is 6 inch square. The driver/motor/lead-screw pitch provide 10000 steps per inch.
The operator must manually insert and remove the square from the platform. The X and Y axis move a drill into the desired position. The Z axis lifts and lowers the drill. The operator presses a switch which tells the motion controller that the square is in place and ready to be drilled. The operator will continuously remove and replace the squares until ready to take a break.
The following is a description of how to set up an OMS board to perform this task.

Procedure: Connect a normally closed switch between user I/O line 0 and ground. This will be the "Ready to Drill" switch.

Enter:

```
AX UU10000          *set up user units so we can reference move to
                     inches
AY UU10000          *10000 steps = 1 inch
AZ UU10000
AX VL.1; AC10;      *set up X axis homing velocity and acceleration
AY VL.1; AC10;      *set up Y axis homing velocity and acceleration
AZ VL.1; AC10;      *set up Z axis homing velocity and acceleration
AX HR AY HR AZ HR   *send each axis to home
AA VL3,3,.5;        *set normal move velocity for X, Y and Z axes
WH                  *start of loop to drill squares indefinitely
                    *(operator removes/replaces square into platform)
    SW0             *wait until operator presses switch
    MA3,3; GO       *move to center of square
    MA,,.5; GO      *move the drill through the square (a 1/2 inch move
                     on the Z axis drills through the square)
    MA,,0; GO       *lift the drill
    MA0,0; GO       *move the platform to home position
WG                  *loop back to starting WH command
(CW)                *operator wants a break so he/she sends CW from
                     keyboard and presses switch once more (since
                     loop will most likely be waiting for the switch at this
                     point)
                    *the loop ends and the following commands exe-
                     cute
MA0,0,0; GO         *move to home position
```

WG WHILE FLAG

The WG command serves as the terminator for the WH command.

QUEUE REQUIREMENTS	
MODE	
AX - AS	2
AA,AM	2
AA/CD	Not valid

Example: (see WH command on the previous page)

CW CLEAR WHILE

The CW command breaks the WH command upon execution of the remaining commands in the loop, i.e. the current execution of the loop is finished. The WH loop is always executed at least one time since the test for the flag is at the bottom.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	1
AA/CD	Not valid

Example: (see WH command on the previous page)

7.10. HOME AND INITIALIZATION CONTROL COMMANDS

These commands allow the initialization of the physical stage with the controller.

HM# HOME

The HM command will cause the current axis to step in the positive direction at the predefined velocity, until the home input line goes true. The position counter will be initialized to the position supplied as a parameter. The velocity should be less than 1024 counts per second to maintain accuracy of the home position loaded. The axis will not stop at home, but will initialize the position counter when the home switch becomes true and decelerates to a stop. The axis may be commanded to go home by following this command with a move absolute to the same position as specified in the HM command. The parameter defaults to zero if none is supplied.

QUEUE REQUIREMENTS		
MODE	**NO ENCODER**	**ENCODER**
AX - AS	4	6
AA,AM	4	6
AA/CD	Not valid	

Example: Find the physical home position of the X axis of the stage. (NOTE: The velocity should be less than 1024 pulses per second to minimize position error for this command.) The motor runs until the home switch input is activated and then initializes the position counter to the parameter supplied. Since the motor decelerates to a stop after reaching home, it is necessary to do an MA# to the same position as specified in the home command if it is desired to physically position the device at home. The following commands will find home, initialize it to 1000 counts, then return to home. In many cases it will not be necessary to return home, only find the position and synchronize the controller to it.

Enter: AX VL1000 HM1000 MA1000 GO

HR# HOME REVERSE

The HR command will cause the current axis to step in the negative direction at the predefined velocity, until the home input line goes true. It behaves exactly like the HM command, except it travels in the reverse direction. The parameter defaults to zero if none is supplied.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AS	4	6
AA,AM	4	6
AA/CD	Not valid	

Example: In a long stage it may be awkward to travel the full distance to home at less than 1024 pulses per second. The following will get close to home at higher speed, then refine the position at lower speed in the reverse direction.

Enter: AX VL100000 HH HM VL1000 HL HR

HA HOME ALL

The HA command flushes the queue, then executes an LM command and an HM0 command on each axis in the AA mode, i.e. all axes will go to limit then home. No parameters are supplied, the position counter will be loaded to zero when the home switch is encountered. The platform connected to the board must eventually be able to trip the limit and home lines on all axes. This command will wait forever until the lines go true at the appropriate times.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AS	Not valid	
AA,AM	4	6
AA/CD	Not valid	

Example: Send all axes to their positive limit then to their home positions.

Enter: AA HA

HJ HOME ADJUST

The HJ command allows the home position to be adjusted to match an individual part. The HA command would first be used to synchronize the SRX position counters with the mechanical stage. This position can then be adjusted for part variation by making an adjustment in position with the joy stick then entering HJ at the desired position. The position counters will be loaded with zero at this position.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AS	Not valid	
AA,AM	Immediate	
AA/CD	Not valid	

Example: Set up all axes on the board so the zero position lines up correctly to the platform.

Procedure: Line up the starting position of the platform using the joystick.

Enter: HJ

KM HOME AND KILL

The KM command will find home and stop generating pulses immediately, i.e. no deceleration ramp will be generated. The position counter is not cleared or reset. Due to motor and platform inertia, the load and board may lose position synchronization.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AS	2	2
AA,AM	Not valid	
AA/CD	Not valid	

Example: Move the Y axis in a positive direction to the home sensor and stop movement as quickly as possible.

Enter: AY KM

KR HOME REVERSE AND KILL

The KR command will find home in reverse and stop generating pulses immediately, i.e. no deceleration ramp will be generated. The position counter is not affected. Due to motor and platform inertia, the load and board may lose position synchronization.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AS	2	2
AA,AM	Not valid	
AA/CD	Not valid	

Example: Move the Y axis in a negative direction to the home sensor and stop movement as quickly as possible.

Enter: AY KR

LM SEEK LIMIT

The LM command will cause the control to travel in the positive direction at the predefined velocity, until the limit switch is encountered, then stop generating pulses immediately.

QUEUE REQUIREMENTS	
MODE	
AX - AS	2
AA,AM	Not valid
AA/CD	Not valid

Example: Make the Z axis run at a velocity of 2000 until it hits the positive limit.

Enter: AZ VL2000 LM

LR SEEK LIMIT IN REVERSE

The LR command will cause the control to travel in the negative direction until the limit switch is encountered, then stop generating pulses immediately.

QUEUE REQUIREMENTS	
MODE	
AX - AS	2
AA,AM	Not valid
AA/CD	Not valid

Example: Make the T axis run at a velocity of 6000 until it hits the negative limit.

Enter: AT VL6000 LR

7.11. MOVE SYNCHRONIZATION COMMANDS

These commands allow the synchronization of moves with external events or multiple axis sequences.

ID INTERRUPT DONE

The ID command will set the done flag and send an "!" to the host. This allows the SRX to signal the host when a string of commands has been completed. In the AA mode, a "!" will be sent to the host when all axes have encounted the ID in its command stream. In the AM mode, an "!" will be returned when the axes active in the most recent move have encountered the ID in the command stream.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	1
AA/CD	Not valid

Example: Interrupt the host CPU after the execution of Move Absolute is finished. When the move is finished the ID command will be encountered in the command queue and will set the done flags.

Enter: AX MA100000 GO ID

II INTERRUPT INDEPENDENT

The II command allows the control to interrupt the host with an "!" when each axis finishes a move. Only those axes which have been supplied a parameter in the most recent move command will cause interrupts.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	1
AA/CD	Not valid

Example: The following command sequence would cause interrupts when the Y and T axes finish. If they do not complete at the same time, two interrupts would be generated.

Enter: MR,1000,,10000; GO II

IN# INTERRUPT NEARLY DONE

The IN command allows the control to interrupt the host with an "!" when the axis or combination of axes is nearly complete. When used in an application involving probing a part after a move, the probes could start accelerating down while the stage is finishing its move, improving the overall system throughput. This command is valid in all modes. The IN command must be entered before the GO or GD command since it is executed before the move is complete. The test is only performed during deceleration. If the IN parameter is greater than the ramp down distance, the interrupt will be generated when the control starts decelerating.

QUEUE REQUIREMENTS	
MODE	
AX - AS	2
AA,AM	2
AA/CD	Not valid

Example: The following sequence would interrupt the host when the X axis is complete and Z axis is within 10,000 counts of being complete. The Y axis completion would be ignored in this example.

Enter:
```
AA
IN0,,10000;
 MR100000,100000; GO
MR,,50000; GO
```

IP INTERRUPT WHEN IN POSITION

The IP command operates like the ID command, except the interrupt is deferred until the stage is within the specified deadband. The GD command should be used in place of the GO command to reset the done flags before the next move. If the position hold HN is not enabled for an axis, the command will behave like an ID command for that axis. This command is available only in models with the encoder option.

QUEUE REQUIREMENTS	
MODE	
AX - AT	1
AU - AS	Not valid
AA,AM	Not valid
AA/CD	Not valid

Example: Send DONE when axis is within deadband.

Enter:
```
AX HV1000 HG100 HD10 HN
MR1000 GO IP (DONE will occur after move is complete and in position.)
```

IC INTERRUPT CLEAR

The IC or the ASCII character Control-Y (hex 19) command is used to clear the done and error flags in the status register and the done flag register, otherwise the axis would always appear to be "done". This command will be executed immediately and will usually be placed in the done and error handler interrupt service routine to clear the interrupt and the associated flags. The Control-Y version of this command is preferred to minimize the latency in its execution. The flags may be polled by an RA or RI command which will also reset the flags.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Clear the flags after an X axis move relative of 5000 steps was flagged as done when an ID executes.

Enter: AX MR5000 GO ID (done flag set) IC

CA CLEAR AXIS DONE FLAG

The CA command operates like the IC command, except it clears the done flag of the addressed axis only.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Not valid
AA/CD	Not valid

Example: After a multi-axis move, clear the Z axis done status only.

Enter: AA MR1000, 2000, 3000, 4000; GO ID
AZ CA

WA WAIT FOR AXES

The WA command, only valid in the AA mode, allows a command to wait until all moves on all axes are finished before it executes.

Some commands which can affect a non-moving axis, such as AN, AF and PA, may execute before a previous move on other axes has finished, especially while in the looping (LS-LE, WH-WG) mode. By preceding these command with a WA, they will not execute until all previously defined moves have finished.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA	2
AA/CD	Not valid

Example: The Z axis auxiliary line controls a laser beam that you only want on while the Z axis moves in a positive direction. The X and Y axes position the laser. You want to repeat the action 10 times.

Enter:
```
AA VL1000,1000,1000; AC10000,10000.10000;
LS10 MR1000,1000; GO WA AN,,1; MR,,500; GO AF,,1;
MR,,-500 GO LE
```

WQ WAIT FOR QUEUE TO EMPTY

The WQ command is a special command that stops the board from processing any new command until the queue for the current axis mode is empty, i.e. all previous moves have finished. This command is not valid in looping (LS-LE, WH-WG) mode.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA	Immediate
AA/CD	Not valid

Example: Move the Y axis 1,000 steps and wait until the move is complete before asking for the position.

Enter:
```
AY MR1000 GO WQ RP
```

SW# SYNC WAIT

The SW command allows synchronization of multiaxes moves or other tasks on one or more SRX boards by using one of the general purpose input lines. This command causes the axes to wait until the general purpose input line has been released (allowed to go high) before proceeding with the next command. The SW command can be used to cause an axis to wait until the others are finished. Wire OR the auxiliary lines from several axes together and connect them to a general purpose input line. Use the SW command on that line. All commands after that will wait until all axes release their auxiliary lines.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	1
AA/CD	Not valid

Example: The following command sequence will cause the X axis move to wait until the Y axis has finished its move and turned off its auxiliary output which has been wired to the general purpose input 0 line.

Enter:
```
AY AN MR2000 GO AF
AX SW0 MR10000 GO
```

The SW command provides a way to synchronize moves on two or more boards. The following example shows one way to do this.

Example: You have 3 eight axes boards, for a total of 24 axes to move together. Call board 1 the "master" and boards 2 and 3 the "slaves". Wire board 1's X axis auxiliary line to the two slave boards' general purpose input 0 line. Send to the master the command "AX PA0",setting the master's X axis auxiliary line low until its move starts. This also sets the slaves' general purpose input 0 line low. Enter the "SW0" command to the two slaves, followed by the move and GO commands. On the master, enter the move command, followed by the GO command. When the master's move starts, the PA command will set the auxiliary line high releasing the wait on the slave boards. All three boards will start their moves.

Procedure: Wire board 1's X axis auxiliary line to board 2's and board 3's general purpose input 0 line.

Enter:
```
(Board 1) AX PA0;
(Board 2) AA SW0; MR200,200,200,200,200,200,200,200; GO
(Board 3) AA SW0; MR300,300,300,300,300,300,300,300; GO
(Board 1) AA MR100,100,100,100,100,100,100,100; GO
```

WT# WAIT

The WT command will wait for the specified number of milliseconds before proceeding with the next command in the queue. In the AA mode all axes will wait. Immediate commands will not "wait". The parameter must be between 1 and 32,000.

QUEUE REQUIREMENTS	
MODE	
AX - AS	3
AA,AM	3
AA/CD	Not valid

Example: You want to produce pulses on the X axis at 5,000 steps/second for 2 seconds, then 10,000 pulses/seconds for 3 seconds, then stop.

Enter: AX JG5000 WT2000 JG10000 WT3000 JG0

7.12. SYSTEM STATUS REQUEST COMMANDS

These commands allow the host to request the status of various move parameters including the status of limit and home switches.

WY WHO ARE YOU

The WY command returns the model type, firmware revision number, and number of controlled axes of the board being addressed, surrounded by line feeds and carriage returns.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

You want to examine the board information.

Enter: WY

Response: <LF><CR>SRX ver 1.75-2<LF><CR>

RP REQUEST POSITION

The RP command returns the current position of the currently addressed axis in the single axis mode or all positions separated by commas in the AA or AM modes. The position will be returned to the host via the data port in ASCII format. This command is not queued, i.e. the current position will be returned immediately even if the axis is in motion. The response is surrounded by line feeds and carriage returns.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: The current position on the Y axis is 12345. Use the RP command to verify the position.

Enter: AY RP

Response: <LF><CR>12345<LF><CR>

RQ REQUEST QUEUE STATUS

The RQ command returns the number of entries available in the queue of the currently addressed axis, in the single axis mode, or all axes separated by commas, in the AA or AM modes. The ASCII string is surrounded by line feeds and carriage returns. The maximum available in each command queue is 200. The response is at a fixed length of 3 characters. For example, if the current free queue space is 67, the response from the board to the RQ command is <LF><CR>067<LF><CR>.

When issuing an RQ command, while defining a contour, the available space in the contouring queue will be returned. The maximum available is 1016. The response is fixed in length at 4 characters.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Immediate

Example: See the size of the command queue for the T axis.

Enter: AT RQ

Response: <LF><CR>200<LF><CR>

BX BIT REQUEST IN HEX

The BX command returns the state of the general purpose I/O bits, in a six digit hex format, surrounded by line feed and carriage return pairs. A one in any binary position signals that bit as being low.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: BX

Response: <LF><CR>810001<LF><CR>
This shows bits 23, 16, and 0 are low, the rest are high.

RA REQUEST AXIS STATUS

The RA command returns the state of the limit and home switches, and the done and direction flags for the currently addressed axis. The done flag is cleared by this command. The status is returned in the following format:

CHARACTER MEANING		
CHAR	**SENT**	**DESCRIPTION**
1	LF	Line feed
2	CR	Carriage return
3	CR	Carriage return
4	P	Moving in positive direction
	M	Moving in negative direction
5	D	Done (ID, II or IN command has been executed, set to N by this command or IC command)
	N	No ID executed yet
6	L	Axis in overtravel. Char 4 tells which direction. Set to N when limit switch is not active.
	N	Not in overtravel in this direction
7	H	Home switch active. Set to N when home switch is not active.
	N	Home switch not active
8	LF	Line feed
9	CR	Carriage return
10	CR	Carriage return

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Not valid
AA/CD	Not valid

Example: The Y axis just encountered a limit, verify its status.

Enter: AY RA

Response: <LF><CR><CR>PNLN<LF><CR><CR>

RI REQUEST INTERRUPT STATUS

The RI command is an AA mode command that returns the same status information on all axes as the RA command in the single axis mode. The 4 character fields for each axis are separated by commas and the string has one line feed and two carriage returns on each end. The done flag is reset by this command.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	Immediate
AA/CD	Not valid

Example: Check the status of a 4 axis board.

Enter: AA RI

Response: <LF><CR><CR>MDNN,MDNN,MDNN,MDNN<LF><CR><CR>

QA QUERY AXIS

The QA command returns the status of the single addressed axis like the RA command except flags are not affected.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Not valid
AA/CD	Not valid

Example: Check the status of the X axis.

Enter: AX QA

Response: <LF><CR><CR>PNNH<LF><CR><CR>

QI QUERY INTERRUPT STATUS

The QI command returns the same information for all axes when in the AA mode, as the QA command does in the single axis mode. The 4 character fields for each axis are separated by commas and the string has one line feed and two carriage returns on each end.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	Immediate
AA/CD	Not valid

Example: Check the status of a four axis board.

Enter: AA QI

Response: <LF><CR><CR>PNNN,MNNN,PDNN,MNLN<LF>CR><CR>

RC REQUEST ACCELERATION

The RC command will return the current acceleration or deceleration of the current axis. This may differ from the programmed acceleration if a cosine (CN) or parabolic (PN) ramp is being generated. When the stage is stopped, the parameter returned will be the acceleration at the beginning of a ramp. When the stage is running at programmed speed, i.e. not accelerating, the parameter returned will be the acceleration at the end of the ramp. While a contour is executing, the value computed to generate the appropriate lead in will be returned. The response to the RC command is surrounded by line feed and carriage return pairs.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Display current acceleration values for all axes on a four axis board.

Enter: AA RC

Response: <LF><CR>2000000,2000000,2000000,2000000<LF><CR>

RV REQUEST VELOCITY

The RV command will return the current velocity at which the axis is moving. This may differ from the programmed velocity if the axis is ramping up to speed or stopping. The response is surrounded by line feed and carriage return pairs. If the JF command is executing, the command only reports the integer part of the velocity.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

Example: Jog the Y axis at 12345 steps per second.
Display the current velocity.

Enter:
```
AY JG 12345
RV
```

Response: <LF><CR>12345<LF><CR>

RU REPORT POSITION IN USER UNITS

The RU command returns the current position in user units (see UU command). The format of response is a floating point number with five characters to the right of the decimal point. This response is surrounded by line feed and carriage return pairs.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Not valid
AA/CD	Not valid

Example: One revolution of a motor is 2000 steps. Define user units so moves can be referenced in revolutions. Move the Z axis 3½ revolutions. Use RU to display the position when the move is complete.

Enter:
```
AZ UU2000; LP0;
MR3.5; GO
(Wait until move is complete.)
RU
```

Response: <LF><CR>3.50000<LF><CR>

7.13. USER UNIT COMMANDS

The following commands allow specification of move parameters in user defined units. The OMS controls will automatically convert all move parameters to these units once they have been initialized.

UU# USER UNITS

The UU command converts all move velocities, distances, etc. to user specified units by multiplying by the specified parameter. This command must be given in the single axis mode but will remain effective in the AA or AM modes. The SRX defaults to user units off at power up or reset. The user units are reset by an RR command, but not reset by an RS command.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Not valid
AA/CD	Not valid

Example: The motor, driver and gear ratio you are using requires 10,000 steps to move one inch. Set up the X, Y and Z axes so you can enter move information in inches.

Enter: AX UU10000 AY UU1000 AZ UU10000

UF USER OFF

The UF command turns off user units. This command is equivalent to and preferred over UU1 since it turns off the mode thus minimizing unnecessary overhead.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Not valid
AA/CD	Not valid

Example: Turn off user unit conversion on X, Y and Z axes.

Enter: AX UF AY UF AZ UF

7.14. POSITION MAINTENANCE COMMANDS

ER#,# ENCODER RATIO

The ER command allows specification of encoder ratio by entering encoder counts, followed by motor counts, for position maintenance mode. These counts must be integers unless user units are enabled. The ratio of encoder counts to motor counts must be equal to one, i.e. encoder counts must match motor counts when slip detection is enabled. All distance, velocity and acceleration parameters are input in encoder counts when this mode is enabled. The correct number of motor counts are generated, while the user need only be concerned with encoder counts. This mode can be combined with user units, allowing units such as inches or revolutions to be specified in encoder counts. All parameters are then input in the user units which have been defined. The ratio defaults to 1 at power up or reset.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AT	Not valid	1
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: You have an encoder connected, through a series of gears, to a stepper motor. When the motor steps 25,000 times, the encoder produces 10,000 counts. Set up an encoder ratio so the hold mode will work correctly.

Enter: ER10000,25000

HV# HOLD VELOCITY

The HV command specifies maximum position hold correction velocity. This is the peak velocity which will be used while making position corrections.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AT	Not valid	2
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: (see HN command on page 7-60)

HG# HOLD GAIN

The HG command allows the user to specify position hold gain parameter. This gain parameter is multiplied by the position error in determing the velocity during correction. The parameter must be between 1 and 32,000. The parameter should be set experimentally by increasing it until the system is unstable, then reducing it slightly below the threshold of stability.

QUEUE REQUIREMENTS		
MODE	**NO ENCODER**	**ENCODER**
AX - AT	Not valid	2
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: (see HN command on the next page)

HD# HOLD DEADBAND

The HD command specifies deadband counts for position hold. If the stage is within this limit, it is considered in position and no further correction will be made. This parameter interacts with the HG command, i.e. a larger deadband will allow a larger gain parameter in many applications. A parameter of zero is allowed.

QUEUE REQUIREMENTS		
MODE	**NO ENCODER**	**ENCODER**
AX - AT	Not valid	2
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: (see HN command on the next page)

HF HOLD OFF

The HF command disables position hold, stall detection and tracking modes. This is the default mode at power up or reset.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AT	Not valid	2
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: Turn off encoder hold mode on the X axis.

Enter: AX HF

HN HOLD ON

The HN command enables position correction after a move and activates the HV, HG and HD commands. Hold and slip detection are disabled if an LP, HM, HR, SA, ST or KL command is entered or if a limit is encountered.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AT	Not valid	2
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: The following commands could be used to set up the position correction mode. This sequence sets up a move velocity of 100,000 steps per second and an acceleration of 500,000 steps per second per second. The position correction velocity is set for 50,000 steps per second, a deadband of 10 steps and correction gain of 2,000. The correction is then enabled. A 200,000 step move is performed, then that position is maintained within the 10 step deadband until commanded to a new position.

Enter:
```
AX VL100000 AC500000
HV50000 HD10 HG2000 HN
MR200000 GO
```

IP INTERRUPT WHEN IN POSITION

The IP command operates like the ID command, except the interrupt is deferred until the stage is within the specified deadband. The GD command should be used in place of the GO command to reset the done flags before the next move. If the position hold HN is not enabled for an axis, the command will behave like an ID command for that axis.

QUEUE REQUIREMENTS	
MODE	
AX - AT	1
AU - AS	Not valid
AA,AM	Not valid
AA/CD	Not valid

Example: Send DONE when axis is within deadband.

Enter: AX HV1000 HG100 HD10 HN
MR1000 GO IP (DONE will occur after move is complete and in position.)

7.15. SLIP AND STALL DETECTION COMMANDS

ES#

The ES command parameter specifies tolerance before slip or stall is flagged in the status register and by the RL command. The mode must be turned on with an IS command and off with an HF command.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AT	Not valid	2
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: Your application can tolerate being up to 5 steps from the desired position before the controlling program should be notified of a slip condition.

Enter: ES5 IS

IS INTERRUPT ON SLIP

The IS command enables the SRX to interrupt the host on slip or stall detection. A "$" will be returned to the host when slip is detected. Hold and slip detection are disabled if an LP, HM, HR, SA, ST or KL command is entered or if a limit is encountered. If a slip occurs, slip detection must be re-enabled.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AT	Not valid	1
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: (see ES command above)

RL RETURN SLIP STATUS

The RL command returns the slip detection status of each axis. An S is returned if slip has occurred for that axis, or else an N is returned. The results are bounded by an LF CR pair, as in other status commands. The number of characters returned corresponds to the number of axes available on the board.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AT	Not valid	Immediate
AU - AS	Not valid	
AA,AM	Immediate	
AA/CD	Not valid	

Example: On a four axis board, see if any axis has slipped.

Enter: RL

Response: <LF><CR>NNSN<LF><CR> (The Z axis has slipped.)

HF HOLD OFF

The HF command disables position hold, stall detection and tracking modes.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AT	Not valid	2
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: Disable slip detection on the X axis.

Enter: AX HF

7.16. ENCODER TRACKING COMMANDS

ET ENCODER TRACKING

The ET command turns on the encoder tracking mode. The axis will track its encoder input, thus allowing one axis to follow the activity of another or a thumbwheel for manual positioning or the movement of another device that produces a signal compatible to the encoder inputs. No acceleration or deceleration ramps are generated. The axis will duplicate the encoder input. The ER command allows the user to scale the motor's movements relative to the encoder.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AT	Not valid	2
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: Set up the X axis so it will follow its encoder input.

Enter: AX ET

HF HOLD OFF

The HF command disables position hold, stall detection and tracking modes.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AT	Not valid	2
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: Turn off encoder tracking on X axis.

Enter: AX HF

7.17. ENCODER HOME CONTROL COMMANDS

HE HOME ENCODER

The HE command enables encoder index mode when an HM or HR command is executed. Home is defined as the logical AND of the encoder index, the external home enable and the encoder quadrant where channel A is positive and channel B is negative. The external enable is low true, i.e. the HH and HL commands are not valid in this mode. The home logic expressed in boolean terms is:

*home= phase_A */ phase_B * index* / home_switch*

QUEUE REQUIREMENTS		
MODE	**NO ENCODER**	**ENCODER**
AX - AT	Not valid	Immediate
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: Set up the Y axis so it will use the encoder signals to recognize the home position.

Enter: AY HE

HS HOME SWITCH

The HS command enables SRX home switch mode to determine where home is when an HM or HR command is executed (default at power up or reset). This mode can also be used with encoders which contain internal home logic by connecting their output to the SRX home input for the appropriate axis. The active level of this input may be controlled by the HH and HL commands.

QUEUE REQUIREMENTS		
MODE	**NO ENCODER**	**ENCODER**
AX - AT	Not valid	Immediate
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: Set up the Y axis so it will ignore the encoder signals and only use the home input to recognize the home position.

Enter: AY HS

7.18. ENCODER STATUS REQUEST COMMANDS

EA ENCODER STATUS

The EA command returns encoder status of the currently addressed axis in the following format:

EA COMMAND RESPONSE DESCRIPTION		
CHAR	**SENT**	**DESCRIPTION**
1	LF	Line feed
2	CR	Carriage return
3	CR	Carriage return
4	E	Slip detection enabled
	D	Slip detection disabled
5	E	Position maintenance enabled
	D	Position maintenance disabled
6	S	Slip or stall detected (reset by execution of EA command)
	N	No slip or stall detected
7	P	Position Maintenance within deadband
	N	Position not within deadband
8	H	Axis is home
	N	Axis is not home
9	N	Unused/reserved
10	LF	Line feed
11	CR	Carriage return
12	CR	Carriage return

QUEUE REQUIREMENTS		
MODE	**NO ENCODER**	**ENCODER**
AX - AT	Not valid	Immediate
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: Examine the status of the Y axis encoder.

Enter: AY RE

Response: <LF><CR><CR>EENPNN<LF><CR><CR>

RE REQUEST ENCODER POSITION

The RE command returns current encoder position of the currently addressed axis in encoder counts. The ASCII string is surrounded by line feed and carriage return pairs.

QUEUE REQUIREMENTS		
MODE	**NO ENCODER**	**ENCODER**
AX - AT	Not valid	Immediate
AU - AS	Not valid	
AA,AM	Not valid	
AA/CD	Not valid	

Example: Examine the current encoder position of the Y axis.

Enter: AY RE

Response: <LF><CR>12345<LF><CR>

7.19. VELOCITY STAIRCASE COMMANDS

The following commands describe the velocity staircase mode. This mode is useful in applications requiring a change in velocity at a prescribed position without stopping.

MP MOVE POSITIVE

The MP command sets the direction logic to move in the positive direction.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	Not valid
AA/CD	Not valid

Example: (see MV command on the next page)

MM MOVE MINUS

The MM command sets the direction logic to move in the negative direction.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	Not valid
AA/CD	Not valid

Example: Set the direction line to move in the minus direction on the Y axis.

Enter: AY MM

MV#,# MOVE VELOCITY

The MV command causes the motor to run to the new absolute position (parameter 1) at the new velocity (parameter 2). When the destination is reached control will be passed to the next command which should be another MV command or an SP command. If the command is not received in time the controller will continue to move at the specified velocity. Note that this is a slave mode and it is the responsibility of the user to provide the commands in time. They may be queued ahead of time. If a new MV command is sent after the controller has already passed the destination specified in the command, the controller will continue to move at the old velocity. Any number of steps can be specified in this manner with both acceleration and deceleration. The controller will not reverse direction if the position has already passed, but will behave as explained above. Thus the direction of the move must be specified before starting the move with the MP or MM commands. All destinations must be in absolute position, no position relative moves are allowed due to the nature of these commands.

QUEUE REQUIREMENTS		
MODE	**NO ENCODER**	**ENCODER**
AX - AS	4	5
AA,AM	Not valid	
AA/CD	Not valid	

Example: Generate a velocity staircase with the breakpoints given in absolute position.

Enter:
```
MP
MV10000,30000
MV20000,50000
MV30000,10000
SP35000
```

The move as shown in Figure 7-2.

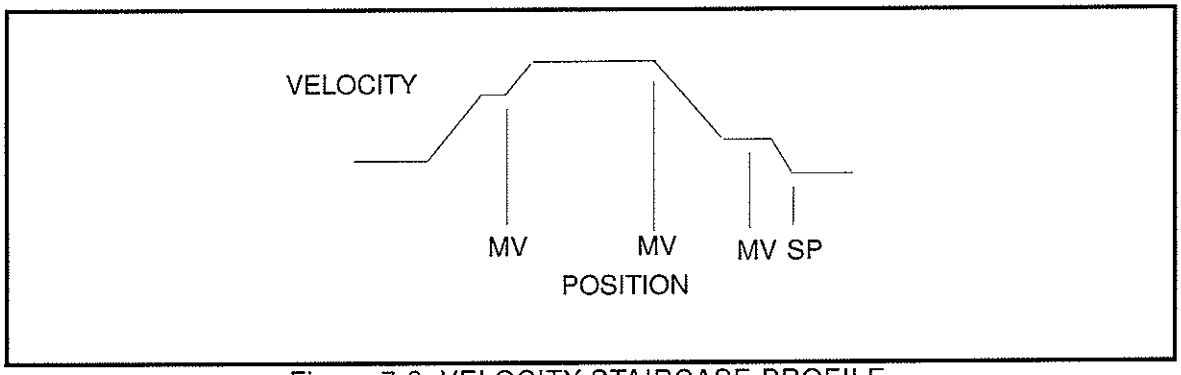


Figure 7-2 VELOCITY STAIRCASE PROFILE

SP# STOP AT POSITION

The SP command will cause the axis to stop at the specified position. The controller will attempt to stop at the specified destination. If there is insufficient distance to stop at the previously specified deceleration when the command is received, the controller will stop as soon as possible at that deceleration. This command is not compatible with the JG.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AS	3	4
AA,AM	Not valid	
AA/CD	Not valid	

Example: (see MV command on the previous page)

FP# FORCE POSITION

The FP command will flush the command queue and attempt to stop at the specified position. The axis will overshoot if there is insufficient distance left to stop at the programmed acceleration.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AS	Flush + 4	Flush + 4
AA,AM	Not valid	
AA/CD	Not valid	

Example: Force axis to stop at 25,000.

Enter: FP25000

7.20. CONSTANT VELOCITY CONTOURING

The SRX will attempt to generate any profile which it is asked to do. It is the responsibility of the host to be sure the acceleration required when generating a circle or any other change in direction is possible within the mechanical constraints of the system. All corners must be defined by arcs and tangents to those arcs, else the change in direction will be instantaneous and generate very large accelerations. The arc radius must be chosen so that the acceleration constraints of the system are met.

AF#,# AUXILIARY OFF

The AF command may be used within a contour definition allowing control of other devices at any instruction within the contour. The AA mode syntax is used. Any auxiliary can be exercised with this command. All axes must be specified or specifically skipped, rather than those axes defined within the contour, as the other commands in this section.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	1
AA/CD	2

Example: (see CD command on page 7-73)

AN#,# AUXILIARY ON

The AN command may be used with a contour by using the AA mode syntax as above. Any auxiliary can be exercised with this command. All axes must be specified or specifically skipped, rather than those axes defined within the contour, as the other commands in this section.

QUEUE REQUIREMENTS	
MODE	
AX - AS	1
AA,AM	1
AA/CD	2

Example: (see CD command on page 7-73)

BL# BIT LOW

The BL command sets the selected general purpose output on (i.e. logic low).

QUEUE REQUIREMENTS	
MODE	
AX - AS	2
AA,AM	2
AA/CD	Not valid

Example: (see the following BH command)

BH# BIT HIGH

The BH command sets the selected general purpose output off (i.e. logic high). The state of general purpose outputs is off at power up or reset.

QUEUE REQUIREMENTS	
MODE	
AX - AS	2
AA,AM	2
AA/CD	Not valid

Example: Set bit 20 high at the start of a contour and low at the end.

Enter:
```
AA CV2000
CD0,0;
BH20
CR0,10000,6.2831853;
BL20
CE
CX
```

CD#,#; CONTOUR DEFINE

The CD command enters contour definition mode. It allows entry of commands for contouring mode. Commands are queued for execution by the CX command. The parameters define the axes for which the contour is defined and the starting position of the contour in absolute units. The contour may be defined on up to 8 axes if circular interpolation is not used or 2 axes with circular mixed with linear interpolation. Attempting to do circular interpolation in a contour which is being defined for more than 2 axes will be flagged as a command error. This command is executed in the AA mode. The contouring axes must be at positions which allow them to reach the specified contouring velocity by the specified position when the contour is executed. If the actual position of the stage is equal to the starting position as defined by the CD command, the stage will jump to the contouring velocity with no ramp up. This could cause the stage to stall if it is not able to accelerate at this high rate. It is recommended that some ramp up distance be allowed. The distance required may be calculated from the equations in Section 1. There is also some ramp down distance as the stage slows from the constant velocity value to a stop. This distance is adjustable using the AC command. It can almost be eliminated using the CK command.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	0
AA/CD	Not valid

Example: The following demonstrates cutting a hole with a 10,000 count radius using constant velocity contouring and circular interpolation. The contouring velocity is set to 1000 pulses per second. A contour is then defined beginning at coordinates 0,0 on the Z and T axes. The auxiliary output of the Y axis is turned on, which could turn on the cutting torch or laser starting the cut at the center of the circle. A half circle is cut from the center to the outside of the hole positioning the cutting tool at the start of the desired hole. The hole is then cut, the torch turned off, the stage stopped and the definition is complete. The stage is then positioned and the hole cut with the CX command. Note that no commas are provided in the MT and CR commands for the inactive X and Y axes within the contour definition. The AN and AF commands must have commas for all axes since they can all be addressed from within the contour definition.

Enter:

```
AA
CV1000 CD,,0,0;
AN,0; CR0,5000,3.1415926
CR0,0,6.2831853
AF,0; MT-10,10000
CE
MT,,-1000,0; GO CX
```

CR#,#,# CIRCULAR INTERPOLATION

The CR command defines a move in a circular pattern from the entry position. The first two parameters are the center of the circle in absolute units and the third parameter is the distance to move in radians. The distance parameter should be supplied to seven significant digits if a full circle is to be generated.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	Not valid
AA/CD	8

Example: (see CD command on page 7-73)

CV# CONTOUR VELOCITY

The CV command allows specification of contouring velocity. It is executed from the AA mode before a contour definition. A contour defined by a CD command cannot be executed if followed by a CV command. Changing this parameter will make any previously defined contours invalid. The contour velocity defaults to 1000 at power up or reset.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	Immediate
AA/CD	Not valid

Example: (see CD command on page 7-73)

CE CONTOUR END

The CE command marks the end of the contour sequence. It will terminate the CD mode, ramp to a stop and exit to the AA command mode when executed. The end of the contour should contain at least a short linear segment just prior to the CE command to initialize the parameters for the deceleration of the stage.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	Not valid
AA/CD	1

Example: (see CD command on the previous page)

CK CONTOUR END and KILL

The CK command will end the contour sequence, like the CE command, except there is no ramp down, i.e. the pulses will stop abruptly. This command should be used with caution to prevent the stage from missing steps or loosing its correct position. It is used in place of the CE command.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	Not valid
AA/CD	1

Example: Same scenario as CD command, but we want to end the contour with the minimum ramp down.

Enter:
```
AA
CV1000 CD,,0,0;
AN,0;CR0,5000,3.1415926
CR0,0,6.2831853
AF,0; MT-10,10000
CK
MT,,-1000,0; GO CX
```

CX CONTOUR EXECUTE

The CX command will execute the previously entered contour sequence. The stage must be positioned such that it can accelerate to speed by the absolute position specified by the CD command it is executing and must be traveling in the proper direction. Once a contour is defined it may be executed at any time by executing a CX command until it is replaced by another contour definition. The CX command cannot be placed within a loop or while construct.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	6
AA/CD	Not valid

Example: (see CD command on page7-73)

MT#,# MOVE TO

The MT command causes the axes defined by the CD command to move to the specified absolute position using linear interpolation. Only the axes being used in a contour must be specified in the contouring mode.

QUEUE REQUIREMENTS			
MODE	LINEAR	PARABOLIC	COSINE
AX - AS	Not valid		
AA,AM	6	30	30
AA/CD	4 + number of axes		

Example: Make a hexagon in CV mode using the X and Y axes.

Enter:
```
AA CV5000;
CD10000,0;
MT20000,0
MT25000,10000
MT20000,2 0000
MT10000,20000
MT5000,10000
MT10000,0
CK
CX
```

RQ REQUEST QUEUE STATUS

The RQ command returns the number of entries available in the contouring queue.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Immediate

Example: Examine contour queue size.

Enter: AA CD0,0; RQ

Response: <LF><CR>1016<LF><CR>

7.21. MACRO CONTROL COMMANDS

MD# MACRO DEFINITION

There are twenty possible macros that can be programmed within the SRX. The MD# (Macro Define) command starts the definition of the macro where # equals 0 through 19. Each macro can contain up to 200 characters and is terminated by a Control-Z. Once the MD# command is entered all characters entered will be placed in the macro. This will continue until a Control-Z is sent to the board. The MX# (Macro Execute) command executes the desired macro. Although macros cannot be nested, they can jump from one to another. After a macro executes it leaves the SRX in the AX mode.

Macros can be executed in the stand alone mode allowing the SRX to function without a host computer or PLC. This is done by entering the appropriate number into the PLC port and then asserting the strobe. Entering the binary equivalent of 100 decimal into the parallel port and asserting the strobe will execute macro 0.

The macros are stored in non-volatile memory and can only be cleared by individually redefining them or by executing an RR (Ram Reset) command. The RR command will erase the macros and set all other parameters to the default settings.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA/AM	Immediate
AA/CD	Not valid

Example: Define macro 8 to set velocities to 20000 and accelerations to 10000 on all axes of an 8 axis board.

Enter:
```
MD8
AA VL20000,20000,20000,20000,20000,20000,20000,20000
^Z
```

MX# MACRO EXECUTE

The MX command causes the specified macro to be executed. Macros cannot be nested but may pass control to another macro. After a macro is executed it leaves the control in the AX mode.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Immediate
AA,AM	Immediate
AA/CD	Not valid

7.22. JOY STICK AND TEACH MODE COMMANDS

The following commands are available in the SRX to support the joy stick and teach mode.

JY JOY STICK MODE

The control enters the joy stick mode when this command is executed. The velocity of the selected axes is under the control of the joy stick while in this mode. Switch B on the joy stick module may be used to select axis pair X-Y, Z-T or U-V for control. The maximum velocity may be changed with the VL command prior to entering the JY command. The actual maximum velocity will be about 70% of the parameter submitted with the VL command. While switch A is on, the control can be taught a position by pressing button C. The first depression of button C after entering JY will remember position 0, then position 1, etc. The JY mode can be exited by entering a KL or SA command from the host or terminal.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	Immediate
AA/CD	Not valid

JZ JOY STICK ZERO

The JZ command allows electrical zeros of the joy stick to be set. While in the JY mode, release the joy stick, allowing it to go to neutral center position, then enter JZ. This will be remembered as the electrical zero until the JZ command is entered again or an RR command is executed.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	Immediate
AA/CD	Not valid

RT# REQUEST TEACH POSITIONS

The RT command returns the position which has been previously taught. Valid numbers are 0 through 99. The format of the returned positions is the same as that returned when an RP is executed in the AA mode.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	Immediate
AA/CD	Not valid

PT# MOVE TO POSITION TAUGHT

The PT command causes the controller to move to the specified position supplied as a parameter. Valid numbers are 0 through 99. The move is a linear move like the MT command.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	Immediate
AA/CD	Not valid

JN# JOY STICK POSITION NUMBER

The JN command allows the sequence of positions being taught to be modified. The parameter supplied is the next position to be taught. Valid numbers are 0 through 99.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AS	Not valid	
AA,AM	Immediate	
AA/CD	Not valid	

JD# JOY STICK DEADBAND

The JD command allows the user to set the joy stick deadband. This establishes the size of the center area of the joy stick movement where no stepping of the motors will occur. The allowed parameter values are 0 through 10.

QUEUE REQUIREMENTS		
MODE	NO ENCODER	ENCODER
AX - AS	Not valid	
AA,AM	Immediate	
AA/CD	Not valid	

JT JOY STICK TEACH MODE

The JT command performs like the JY command while switch A is on. The serial port is not active but commands may be entered through the PLC port from a PLC or switch register. The commands allowed through the PLC port are as follows:

COMMAND	DESCRIPTION
0 to 99	Equivalent to a JN command with the same parameter.
100 to 127	Not valid

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	1
AA/CD	Not valid

JR JOY STICK RUN MODE

The JR command allows the positions to be commanded through the PLC port. This command is active while switch A is off. The commands allowed through the PLC port are as follows:

COMMAND	DESCRIPTION
0-99	Equivalent to a PT command with the same parameter.
100-119	Execute macro 0 through 19 respectively. The control exits a macro in the AX mode.
120	Execute an HA command.
121	Execute an HJ command.
122-126	Not valid
127	Execute a KL command, exiting JR mode and returning control to the serial port.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	Immediate
AA/CD	Not valid

PE#,# POSITION ENTER

The PE command allows positions to be entered or modified without moving the stage. The first parameter is the position number followed by the desired positions. Entering a comma as a place holder with no position parameter will leave that axis position unchanged allowing only the desired axis to be modified to any position.

QUEUE REQUIREMENTS	
MODE	
AX - AS	Not valid
AA,AM	Immediate
AA/CD	Not valid

Example: Since the joy stick cannot be used on the R and S axes, values for them must be entered using the PE command. Add positions for R to be 10000 and S to be -24680, to position number 17. The other axes positions will remain unaltered.

Enter: PE17,,,,,,,10000,-24680

7.23. COMMAND SUMMARY

The following commands are included in the SRX family of motor controllers. The '#' indicates a signed integer input parameter or a signed fixed point number of the format ##.# when user units are enabled. With User Units enabled, distances, velocity and acceleration parameters may be input in inches, revolutions, etc.

SUMMARY OF COMMANDS IN CHAPTER 7		
COMMAND	**SECTION PAGE NUMBER**	**COMMAND DESCRIPTION**
AA	3	Any following commands are for the AA (All Axes) mode
AC#	20	Acceleration, set acceleration/deceleration register
AF	16, 71	Auxiliary off
AM	3	Axes multitasking mode
AN	15, 71	Auxiliary on
AR	6	Any following commands are for the R axis
AS	6	Any following commands are for the S axis
AT	5	Any following commands are for the T axis
AU	5	Any following commands are for the U axis
AV	5	Any following commands are for the V axis
AX	4	Any following commands are for the X axis (default on reset)
AY	4	Any following commands are for the Y axis
AZ	4	Any following commands are for the Z axis
BH#	18, 72	Set selected I/O bit high (off)
BL#	18, 72	Set selected I/O bit low (on)
BX	19, 52	Return bit status in hex format
CA	47	Clear done flag of currently addressed axis
CD#,#;	73	Define a contour
CE	74	End contour definition, ramp to a stop
CK	74	End contour definition, immediately stop step pulses
CN	11	Cosine on, enable cosine velocity profiles
CR#,#,#	75	Circular interpolation, move in a circle
CV#	75	Contouring velocity, definition
CW	39	Clear while flag, i.e. terminate WH/WG loop
CX	76	Contour execute
EA	66	Encoder status, return encoder status of currently addressed axis
EF	7	Echo off, turn off echo to host (default on power on)
EN	7	Echo on, turn on echo to host
ER#,#	58	Encoder ratio, set encoder count to motor count ratio
ES#	62	Encoder slip tolerance, set tolerance before slip or stall is flagged
ET	64	Encoder tracking, set encoder tracking mode

SUMMARY OF COMMANDS IN CHAPTER 7		
COMMAND	**SECTION PAGE NUMBER**	**COMMAND DESCRIPTION**
FP#	70	Force position, flush queue and attempt to stop at specified position
GD	30	Go and reset done flags
GO	29	Go command, start execution of motion
HA	41	Home all, execute an LM then an HM0 for all axes
HD#	59	Hold deadband, specify deadband tolerance for position hold
HE	65	Encoder home mode, set home on encoder logic
HF	60, 63, 64	Hold off, disable position hold, slip detection and tracking modes
HG#	59	Hold gain, specify position hold gain parameter
HH	8	Home high, home switches are active high
HJ	42	Home adjust, set the current position of each axis to 0
HL	8	Home low, home switches are active low
HM#	40	Home, find home and initialize the position counter
HN	60	Hold on, enable position correction after move
HR#	41	Home reverse, find home in reverse direction and initialize position counter
HS	65	Home switch, enable home switch mode
HV#	58	Hold velocity, specify maximum position hold correction velocity
IC	47	Interrupt clear, clear done status
ID	45	Interrupt host when done and set done flag
II	45	Interrupt independent
IN#	46	Interrupt when nearly done
IP	46,61	Interrupt when in position
IS	62	Interrupt slip, interrupt host on slip or stall detection
IT	14	Initialize default velocity and acceleration parameters
JD#	81	Joy Stick dead band, a value from 0 to 10
JF#	31	Jog the current axis at fractional rates
JG#	31	Jog command, run motor at specified velocity until a new velocity command is sent or it is stopped by a stop or kill command
JN#	80	Joy stick position number, reset the sequence of positions being taught
JR	82	Joy stick run mode, simulate the stand alone run mode as long as switch A is off
JT	81	Joy stick teach mode, simulate the stand alone teach mode as long as switch A is on
JY	79	Enter joy stick and teach mode
JZ	79	Electrically zero the joy stick
KL	64	Kill, flush queue and terminate pulse generation immediately on all axes without decelerating

SUMMARY OF COMMANDS IN CHAPTER 7		
COMMAND	**SECTION PAGE NUMBER**	**COMMAND DESCRIPTION**
KM	42	Home and kill pulse generation
KR	43	Home in reverse and kill pulse generation
LE	36	Loop end, terminate most recent LS command
LF	9	Disable limit switches for selected axis
LM	43	Seek limit in positive direction
LN	9	Enable limit switches for selected axis
LP#	23	Load position, load position counter with parameter
LR	44	Seek limit in negative direction
LS#	35	Loop start, set loop counter, from 1 to 32000 loops; (may be nested to 4 levels)
MA#	24	Move absolute, move to absolute position
MD#	78	Macro define, allow user to combine several SRX commands into one of 20 macros
ML#,#;	26	Move linear, move specified distance relative from current position
MM	68	Move minus, set minus direction for MV type move
MO	27	Move one pulse in current direction
MP	68	Move plus, set positive direction for MV type move
MR#	25	Move relative, move specified distance from current position
MT#,#;	27, 76	Move to, move to specified absolute position in a linear move
MV#,#	69	Move velocity, move to first parameter (absolute position) at second parameter velocity without stopping at end of move
MX#	78	Execute macro #
PA#	17	Power automatic, turn power on before each move and off after the move
PE#,#	82	Position enter, allow positions to be entered and edited
PF	12	Parabolic off, disable parabolic ramps, i.e. linear ramps will be generated
PN#	11	Parabolic on, enable parabolic ramps
PT#	80	Move to position previously taught
QA	54	Query status of switches and flags for addressed axis without affecting flags
QI	55	Query status of switches and flags on all axes without affecting flags
RA	53	Return status of switches and flags and reset flags
RB	19	Return programmed direction of I/O bits in hex format
RC	55	Return current acceleration or deceleration of the current axis
RE	67	Request encoder position, return current encoder position

SUMMARY OF COMMANDS IN CHAPTER 7		
COMMAND	**SECTION PAGE NUMBER**	**COMMAND DESCRIPTION**
RI	54	Return status of switches and flags for all axes and reset flags
RL	63	Return slip status of each axis
RM#	28	Return remainder of position divided by parameter in position counter
RP	51	Request position, return current position
RQ	52, 77	Request queue status, return number of queue entries available
RR	12	Reset default parameters (all RAM) and restart
RS	13	Software reset of SRX
RT#	80	Return position previously taught
RU	56	Return current position in user units
RV	56	Return current velocity at which the axis is moving
SA	33	Stop all, flush queue and stop all axes with deceleration
SB#	13	Set baud rate to parameter supplied
SD	34	Stop all axes and clear any done flags
SE#	17	Set settling time before power is reduced in PA mode
SF	10	Soft limit off, restore normal overtravel operation
SL	10	Soft limit mode, allow pulse train to ramp down on overtravel
SP#	70	Stop at position, stop at specified position if possible after all commands have been executed
ST	33	Stop, flush queue and decelerate to stop
SW#	49	Sync wait, wait for the input bit to be released by other controllers
UF	57	User units off, turn off user unit translation
UU#	57	User units, multiply acceleration, velocity and distance parameters by specified parameter
VB#	22	Base velocity, set base velocity
VL#	21	Set maximum velocity to be used in profile
VS#,#,#	32	Velocity stream, slave velocity mode for profiling
WA	48	Wait until all moves on all axes are finished
WD	37	While end, WS loop terminator
WG	39	Terminate WH loop
WH	38	While, execute all commands until WG loop terminator, until flag cleared by CW command
WQ	48	Wait until current axis queue is empty
WS#	37	While sync, execute while sync is true
WT#	50	Wait, wait for specified number of milliseconds
WY	51	Who are you, return model and software revision

8.
HOST SOFTWARE

8.1. INTRODUCTION

The following describes the Oregon Micro Systems' program SRX.EXE which is available on the demo disk shipped to first time users of SRX and SSX family products. The C source code for the program is also on the demo disk. No BIOS or assembly language routines are used. Other files on the demo disk are OMS command files showing some of the features of the SRX/SSX program.

The program SRX.EXE is a simple terminal emulation program to allow a user of an Oregon Micro Systems SRX/SSX board to communicate with the board using the RS-232 compatible COM1 or COM2 serial ports on a PC/XT/AT compatible computer.

This program allows the user to use either serial port to send commands to and receive responses from an SRX or SSX board at baud rates from 300 to 19200, with hardware handshaking. OMS commands can optionally be sent to the board as an ASCII text file.

8.2. COMMAND LINE OPTIONS

There are several command line options that can be used by the program. They are:

/F:<filename>	This allows a file of OMS commands to be sent to the board immediately at program start.
/B:<baud rate>	Sets the baud rate at which the program will communicate with the board. Valid baud rates are 300, 600, 1200, 2400, 4800, 9600, and 19200. The default is 9600.
/P:<comm port #>	Establishes which comm port the program will use to communicate with the board. Valid values are 1 and 2. The default is 1.

8.3. SPECIAL COMMAND FILE CHARACTERS

There are several special characters that can be included in an OMS command file that alter the way the program downloads the command file to the controller. They are:

?	The question mark causes the program to stop sending the command file to the board until a done flag is received from the board. This mode can be toggled on or off by using the Shift-F5 key. The '?' must be preceded by an ID command otherwise the program will wait forever for a DONE that was never programmed.
&	The ampersand causes the program to send a command file whose name immediately follows it, to the board. This mode only works if

the first file was loaded at program start using the /F: command line option. The '&' allows several files to be chained together.

* The asterisk allows comments to be imbedded into a command file. Every character following the '*' up to the next new line will be printed to the screen and not be sent to the controller.

$ The dollar sign in a command file will pause the download process until a key on the keyboard is pressed.

8.4. FUNCTION KEY DESCRIPTIONS

The function keys have special uses. The following list describes them.

F1 Display help information about program operation. This shows three screens of information on program features and settings.

F2 Erase the screen.

F3 Change the baud rate. The board's baud rate should be changed using the SB command before you change the program's baud rate. You are prompted to enter the desired baud rate.

F4 Show the current directory using the "DIR /W" command.

F5 Send a file to the board. The program prompts you for a file name. If you don't enter one and press return, the previously sent file will be sent again.

F6 Temporarily create a DOS shell. You can then enter DOS commands at the DOS prompt without exiting the program. Type EXIT to leave the DOS shell and return to the program and the <SRX> prompt.

F7 Use printer port to send PLC (programmable logic controller) commands to the controller. If you have a printer port, LPT1, at base address 378 hex, you can use it to simulate a PLC device when connected to the PLC connections of the SRX/SSX board. You need to build a cable with connections as indicated in Table 8-1:

You are prompted to enter a PLC code as a decimal number from 0 to 127. Any number greater than 127 will exit the PLC mode. The board will, of course, have to be in the PLC mode before this program option will work. You can use the JR command to make the board enter the PLC mode before you press F7.

F8 Display controller's position memory. All 100 positions will be displayed, ten at a time. After a group of 10 are shown, the display pauses, allowing the user to continue or exit.

F9 Toggle program echo mode. If program echo is turned on, everything typed on the keyboard is displayed. If the controller's echo is also on, you will see two of every character you type. Either press the F9 key or enter the EF command to disable one of the echoes.

F10 Program a macro into the controller. This key allows limited editing of the text being programmed into the macro before it is actually sent to the board. You are prompted to enter the macro number you want

to program. Some instructions are displayed to describe how this routine works. Then you enter the commands you want to store into the macro. Press ESC to exit this mode.

Shift-F1 Send XOFF char to controller. This is for future use. Currently the controllers do not recognize the XON-XOFF software hand shaking.

Shift-F2 Send XON char to controller. This is for future use. Currently the controllers do not recognize the XON-XOFF software hand shaking.

Shift-F3 Set DTR (pin 20) high (true). Sets the RS-232 pin high immediately, bypassing all interrupt routines.

Shift-F4 Set DTR (pin 20) low (false). Sets the RS-232 pin low immediately, bypassing all interrupt routines.

Shift-F5 Toggle 'Wait for done' mode. This disables/enables the ability of the program to pause the download process until the controller sends the done flag character.

Table 8-1 PIN DEFINITIONS FOR THE PRINTER PORT

LPT1 PORT PIN #	SIGNAL DIRECTION	SRX/SSX J7 PIN#
1 (STROBE)	——>	A32 (I/O BIT 7)
2 (BIT 0)	——>	A25 (I/O BIT 0)
3 (BIT 1)	——>	A26 (I/O BIT 1)
4 (BIT 2)	——>	A27 (I/O BIT 2)
5 (BIT 3)	——>	A28 (I/O BIT 3)
6 (BIT 4)	——>	A29 (I/O BIT 4)
7 (BIT 5)	——>	A30 (I/O BIT 5)
8 (BIT 6)	——>	A31 (I/O BIT 6)
11 (BUSY)	<——	C32 (I/O BIT 23)
23 (GROUND)	<—>	C24 (GROUND)

8.5. OTHER PROGRAM FEATURES

When this program is run, it will create a prompt like this: <SRX>.

To exit the program press Control-C or ALT plus any key. You are then prompted to enter another key. If the key is Esc, K or k (kill), or a SPACE, the program will send a KL command to the board stopping motor movement. If the key is S or s (SAVE), all previous commands entered from the keyboard will be saved to a file called history.srx in the current directory. If the key is E or e (EXIT) the program will exit without saving anything. Any other key will return the program to normal operation.

Beside holding down the CONTROL key and pressing an alpha key, a control character may also be sent to the board by using the CARAT ^ (ASCII 94) symbol in front of the alpha character for the control character. This program translates the pair to the single control character. For example, to send a Control-Z (ASCII 26) send a ^ (ASCII 94) followed by a

Z (ASCII 90). This can be done from a command file as well as from the keyboard. With this feature, macros can be terminated from a file by using the '^' 'Z' combination which will not be seen by DOS as the END-OF-FILE character that a CONTROL-Z is.

Whenever the board is reset or its input buffers are full, it will release the CTS line. This will make a message appear on the screen, "CTS line is inactive, waiting for it to go active...". You will not be able to send anything else to the board until it releases the line. The program will then display the line, "CTS line is now active, sending data to board...".

This program decodes the special status characters the board sends to signal status changes as shown in Table 8-2:

Table 8-2 SPECIAL STATUS CHARACTERS

CHARACTER	MESSAGE DISPLAYED
#	"Program detects COMMAND ERROR FLAG received from SRX board"
$	"Program detects ENCODER SLIP FLAG received from SRX board"
@	"Program detects LIMIT SWITCH ACTIVE FLAG received from SRX board"
!	"Program detects ALL AXES DONE FLAG received from SRX board"

9. SERVICE

9.1. USER SERVICE

The SRX family of controllers contain no user serviceable parts.

9.2. THEORY OF OPERATION

The 68000 microprocessor on the SRX controllers maintains four concurrent processes. The highest priority process calculates the desired pulse frequency 1024 times each second with a proprietary algorithm (patent number 4,734,847). This frequency is fed to U53, U55, U57 and U59 which generate the pulse trains. The velocity profile and synchronization of each axis is also handled by the 68000.

The commands from the host computer are temporarily stored in a 124 character buffer until the 68000 microprocessor can parse them. The command is then executed immediately or routed to separate command queues for each axis. The command queue contains a list of addresses to execute followed by an optional parameter. A command from the host may be expanded into several commands to the appropriate axis. The GO command, for example, will expand into start, ramp up, constant velocity and ramp down commands. The LS command will save its parameter, i.e. the loop count, on a loop stack along with the address of the LS command to be used by the next LE command as a target for a jump command. The LE command will decrement the loop count and jump to the most recent LS command providing the loop count has not reached zero. If the loop count has reached zero and it is not nested inside another loop, the queue space will be flagged as available and the next instruction in the queue will be executed.

APPENDIX A. LIMITED WARRANTY

The Seller warrants that the articles furnished are free from defect in material and workmanship and perform to applicable, published Oregon Micro Systems, Inc. specifications for one year from date of shipment. This warranty is in lieu of any other warranty express or implied. In no event will Seller be liable for incidental or consequential damages as a result of an alleged breach of the warranty. The liability of Seller hereunder shall be limited to replacing or repairing, at its option, any defective units which are returned f.o.b. Seller's plant. Equipment or parts which have been subject to abuse, misuse, accident, alteration, neglect or unauthorized repair are not covered by warranty. Seller shall have the right of final determination as to the existence and cause of defect. As to items repaired or replaced, the warranty shall continue in effect for the remainder of the warranty period, or for 90 days following date of shipment by Seller of the repaired or replaced part whichever period is longer. No liability is assumed for expendable items such as lamps and fuses. No warranty is made with respect to custom equipment or products produced to Buyer's specifications except as specifically stated in writing by Seller and contained in the contract.

APPENDIX B. TECHNICAL SUPPORT

Oregon Micro Systems, Inc. can be reached for technical support by any of the following methods:

1. Internet E-Mail:	support@OMSmotion.com
2. World Wide Web:	http://www.OMSmotion.com
3. Telephone	8:00 a.m. - 5:00 p.m. Pacific Standard Time (503) 629-8081
4. Facsimile:	24 Hours (503) 629-0688
5. USPS:	Oregon Micro Systems Inc 1800 NW 169th Place Suite C100 Beaverton OR 97006

RETURN FOR REPAIR PROCEDURES

1. Call Oregon Micro Systems, Inc. Customer Service at 503-629-8081.

2. Explain the problem and we may be able to solve it on the phone. If not, we will give you a Return Materials Authorization (RMA) number.

   Mark the RMA number on the shipping label, packing slip and other paper work accompanying the return. We cannot accept returns without an RMA number.

3. Please be sure to enclose a packing slip with the RMA number, serial number of the equipment, reason for return, and the name and telephone number of the person we should contact if we have further questions.

4. Pack the equipment in a solid cardboard box secured with packing material.

5. Ship prepaid and insured to:

   OREGON MICRO SYSTEMS, INC.
   Twin Oaks Business Center
   1800 NW 169th Place, Suite C100
   Beaverton, OR 97006

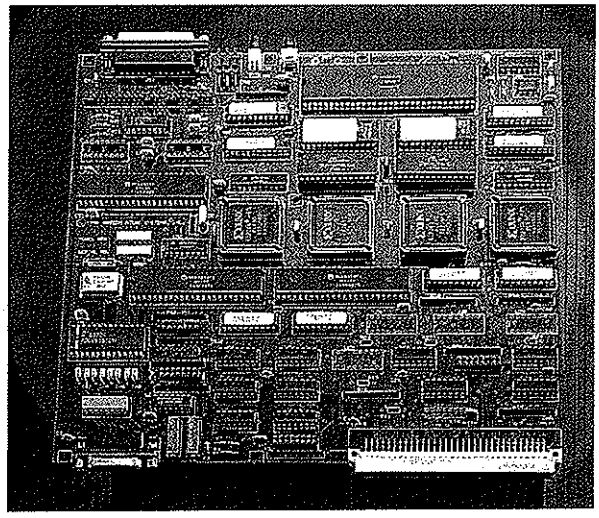
APPENDIX C. SPECIFICATIONS

OREGON MICRO SYSTEMS, INC.

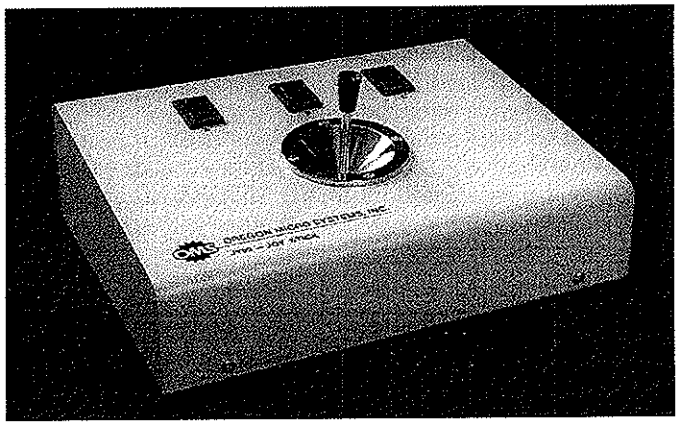
SRX
INTELLIGENT MOTOR CONTROLLERS FOR RS232 and RS422 COMPATIBLE COMPUTERS, PLCs and STAND ALONE

FEATURES

- Up to EIGHT axes on one control card
- 32 bits of user definable I/O available
- Functions as an intelligent peripheral to RS232 or RS422 compatible computers
- Compatible with most PLCs
- Operates stand alone with joy stick teach mode
- User definable macros allow predefined command sequences for stand alone operation
- Uniform pulse train for minimum torque ripple
- Minimal move latency
- Watchdog timer
- Circular interpolation on any 2 axes
- Linear interpolation at constant velocity on up to eight axes
- Cosine (S-curve) and parabolic velocity profile
- Encoder feedback available as option on up to 4 axes
- Automatic conversion to user defined units, i.e. inches or revolutions, simplifies software
- Smooth, high resolution, resonance free motion
- Compatible with any stepping, servo or linear motor drive that accepts digital step and direction signals
- Synchronized or independent motion on any or all axes
- Crystal controlled pulse rates from 0 to 522,000 steps per second
- Independent limit and home switches for each axis
- 68000 microprocessor for velocity profile synthesis and high level command interpretation
- Supports resolution of 0.0072 degrees per step (0.126 milliradians per step) or 50,000 steps per revolution (function of motor and driver)
- Position range of 134 million pulses on each axis
- All data and commands are programmed in ASCII command strings
- Standard eurocard format
- 1 year limited warranty



SRX-8, Eight axes controller



JY02-S, Joy stick position control with teach mode

DESCRIPTION

The SRX intelligent motion controller allows control of up to 8 axes on one controller card. It can be used with computers or terminals with RS232 or RS422 serial communication compatibility, with a PLC or as a stand alone controller. All commands are passed through the serial port when used with an RS232 or RS422 compatible computer or through a 7 bit parallel port when used with a PLC. A joy stick interface is built into the SRX. It can be taught up to 100 positions using up to 6 axes in the teach mode. These positions are retained even with power off in the non-volatile RAM. Each axis has a separate command queue allowing the host computer to transfer a command string then proceed with other tasks, while the SRX manages the motion process. Each axis can perform individual unrelated moves or they can be coordinated as required by the application.

The SRX generates step and direction pulses for control of most popular step motor drivers. It also supports servo or linear motor controllers which accept step and direction inputs. The SRX supports high resolution microstepping of 50,000 steps per revolution with a standard 200 step per revolution (1.8 degree per step) stepping motor by developing the high pulse rates required for these applications. This high resolution allows the stepping motor to run smoothly at all speeds and minimizes low speed torque loss due to mechanical resonance effects. Constant velocity contouring with circular interpolation on any 2 axes and linear interpolation on up to eight axes at constant velocity are available for machining applications.

Simple ASCII commands can be easily sent to the board from any high level language, which allows input and output to an I/O device, for example Basic, Pascal, and C. An additional 32 bits of general purpose input and output lines can be used to monitor or initiate other events and are under the control of the host computer. Complex move sequences, time delays, status checks and control of other external events can be programmed through the serial interface.

The SRX can operate without a host computer or PLC in stand alone mode by teaching positions with the joy stick then executing predefined command sequences. These command sequences may be input through the parallel port from a PLC or manually via a switch register. User defined macros simplify input command sequences and allow complex unattended operations in the stand alone mode. The SRX can execute any user defined macro at power up or reset without commands from any external source.

Incremental encoder feedback is available as an option on up to 4 axes for those applications requiring precise position feedback and/or correction. The encoder option can correct for position errors, monitor for slip or stall, or allow tracking of one motor with another.

PROGRAMMING

The OMS motion controller is easily programmed with double character ASCII commands through an extensive command structure. These commands are combined into character strings to create sophisticated motion profiles. It includes a 200 command and parameter buffer for each axis and a command loop counter which allows multiple executions of any command string. Twenty macros of 200 characters each may also be taught to simplify repetitive tasks or used in stand-alone and PLC control modes.

The following commands are available in the SRX family of motion controllers. Some of the commands expect a numerical operand to follow. These commands are identified with a '#' after the command. The '#' indicates a signed integer input parameter or a signed fixed point number of the format ##.# when user units are enabled. With user units defined, distances, velocity and acceleration parameters may be input in inches, revolutions, etc.

Synchronized moves may be made by entering the AA command. This command performs a context switch which allows entering commands of the format MRx#,y#,z#, t#,u#,v#,r#,s#;. Numbers are entered for each axis which is to be commanded to move. An axis may be skipped by entering the comma with no parameter. The command may be prematurely terminated with a ";", i.e. a move requiring only the X and Y axes would use the command MRx#, y#; followed by the GO command. Each axis programmed to move will start together upon executing the GO command. The SRX can be switched back to the unsynchronized mode by entering the desired axis command such as AX.

The following summarizes the SRX command set:

AXIS SPECIFICATION COMMANDS

The following commands specify the axis to which the commands are to be directed. They remain in effect until replaced by another command of the same type.

AA AXIS ALL

The AA command will perform a context switch to the synchronized mode.

AM AXES MULTITASKING

The AM mode allows several tasks to be managed simultaneously. For instance, a task may be performing coordination motion on 2 axes, while a second task is performing unrelated but simultaneous motion on another axis.

AX AXIS X

The AX command directs all the following commands to the X axis.

AY AXIS Y

The AY command directs all the following commands to the Y axis.

AZ AXIS Z

The AZ command directs all the following commands to the Z axis.

AT AXIS T

The AT command directs all the following commands to the T axis.

AU AXIS U

The AU command directs all the following commands to the U axis.

AV AXIS V

The AV command directs all the following commands to the V axis.

AR AXIS R

The AR command directs all the following commands to the R axis.

AS AXIS S

The AS command directs all the following commands to the S axis.

SYSTEM CONTROL COMMANDS

These commands allow control of various system parameters and operating modes to allow the user to optimize the response of the system for his/her application needs.

EN ECHO ON

The EN command enables echoing from the SRX.

EF ECHO OFF

The EF command disables echoing from the SRX.

HH HOME HIGH

The HH command sets the sense of the home switch on the current axis to active high.

HL HOME LOW

The HL command sets the sense of the home switch on the current axis to active low.

LF LIMITS OFF

The LF command turns off the limit switches for the addressed axis. This allows the stage to move beyond the limit switch and should be used with caution.

LN LIMITS ON

The LN command restores the operation of the limit switches for the addressed axis.

SL SOFT LIMIT

The SL command changes the operation of the limit inputs causing the output pulse train to ramp down instead of terminating immediately. The output queue is not flushed except for the current move.

SF SOFT LIMIT OFF

The SF command restores the normal operation of the limit switches.

CN COSINE ON

The CN command enables cosine velocity ramps, i.e. half sinusoidal acceleration profiles for all axes.

PN# PARABOLIC ON

The PN command enables parabolic ramps. The parameter selects the point of truncation.

PF PARABOLIC OFF

The PF command restores linear acceleration and deceleration ramps.

RR RESET AND INITIALIZE RAM

The RR command initializes all parameters to their default values then causes a hardware reset. All parameters including macros, velocity, acceleration, teach positions and baud rate are initialized to factory defaults.

RS RESET

The RS command is a software reset which causes the SRX microprocessor to reset. All programmable values are set to factory defaults.

SB# SET BAUD RATE

The SB command sets the communication baud rate to the rate specified. The SRX is initialized at the factory to 9600 baud, 8 data bits and no parity. If the rate is changed with this command it will remain until changed by another SB command or reset to factory default with the RR command. Rates of 300, 600, 1200, 2400, 4800, 9600 and 19200 bits per second are supported.

IT INITIALIZE DEFAULTS

The IT command initializes the control to the default velocity and accelerations.

MOVE SPECIFICATION COMMANDS

These commands allow specification of move parameters. They allow move parameters to be tailored to the user's system requirements.

AC# ACCELERATION

The AC command sets the acceleration/deceleration value. This value is used to establish the rate of acceleration and deceleration when a move command is invoked.

VL# VELOCITY

The VL command sets the maximum velocity value of the axis being programmed. The value is used to establish the maximum number of pulses per second sent to the motor driver from the SRX board when one of the move execution commands is invoked.

VB# VELOCITY BASE

The VB command allows the velocity ramp to start at the specified velocity. This allows faster acceleration and the ability to pass through resonance quickly in some applications.

LP# LOAD POSITION

The LP command will immediately load the position supplied as a parameter into the absolute position register of the axis.

MA# MOVE ABSOLUTE

The MA command will set up the axis to move to the absolute position supplied as a parameter.

MR# MOVE RELATIVE

The MR command will set up the axis to move relative from the current position at the time the move is executed.

ML#,# MOVE LINEAR

The ML command uses linear interpolation to perform a straight line relative move to the new location. Up to 8 axes may be moved together in the AA or AM modes.

MT#,# MOVE TO

The MT command uses linear interpolation to perform a straight line move to the specified absolute position. Up to 8 axes may be moved together in the AA or AM modes.

MO MOVE ONE PULSE

The MO command will output one step pulse in the current direction without the latency of a ramp up to velocity. This command is not available in models with an encoder option.

RM# REMAINDER

The RM command will divide the position counter by the parameter supplied and replace the position counter with the resulting remainder as an absolute position. This command is useful in continuously rotating axis applications.

MOVE EXECUTION COMMANDS

These commands allow execution of the moves which have been previously specified.

GO GO

The GO command will initiate the move which has been previously programmed with such commands as MA, MR, MT, and ML.

GD GO AND RESET DONE FLAG

The GD command resets the done flags on the active axes then proceeds with the move identical to the GO command.

JG# JOG

The JG command is a velocity command and will jog the axis at the velocity supplied as a parameter. The velocity may be changed without stopping by entering another JG command.

JF#,# JOG FRACTIONAL VELOCITIES

The JF command will jog the current axis at fractional rates.

VS#,#,# VELOCITY STREAMING

The VS command will generate a pulse train without acceleration or deceleration at the rates specified for the time specified by the first parameter.

MOVE TERMINATION COMMANDS

The following commands allow termination of move sequences in process.

ST STOP

The ST command flushes the queue for the currently addressed axis only and causes the axis to decelerate to a stop at the rate previously specified in an AC command.

SA STOP ALL

The SA command flushes all queues and causes all axes to decelerate to a stop at the rate previously specified in an AC command.

SD STOP AND RESET DONE

The SD command will stop all axes and clear any done flags.

KL KILL

The KL command will flush the command queue and terminate pulse generation of all axes immediately.

LOOP CONTROL COMMANDS

These commands allow move sequences to be repeated within loops. Loops can be nested up to four levels deep on each axis.

LS# LOOP START

The LS command sets the loop counter for the axis being programmed. The parameter specifies the number of times the loop will be executed. Loops may be nested up to 4 levels deep.

LE LOOP END

The LE command terminates the most recent LS command.

The following commands can be used to synchronize multiple SRX boards or synchronize them to external events:

WS# WHILE SYNC TRUE

The WS command will execute the commands between the WS and WD commands as a loop, while the general purpose input line is true, i.e. low. The test is at the bottom of the loop and thus will always be executed at least once.

WD WHILE END

The WD command serves as the loop terminator for the WS command.

WH WHILE

The WH command will execute all commands between it and the terminating WG command as a loop until terminated by a CW command. This allows indefinite loops to be terminated by the host computer.

WG WHILE FLAG END

The WG command serves as the terminator for the WH command.

CW CLEAR WHILE

The CW command terminates the WH command sequence upon execution of the next WG instruction, thus the loop is always executed at least once.

HOME AND INITIALIZATION CONTROL COMMANDS

These commands allow the coordination of the physical stage home position with the SRX position register.

HM# HOME COMMAND

The HM command will find home and initialize the position counter to the position supplied as a parameter.

HR# HOME REVERSE

The HR command will find home in the reverse direction and initialize the position counter to the position supplied as a parameter.

HA HOME ALL

The HA command executes an LM command then an HM0 command, i.e., all axes will go to the limit switch then home. No parameters are supplied; the position counter will be loaded to zero when the home switch is encountered. This command will flush the queues if executed from the serial port.

HJ HOME ADJUST

The HJ command allows the home position to be adjusted to match an individual part. The HA command would first be used to synchronize the physical stage with the SRX position counter. This position can then be adjusted to compensate for part variations by adjusting with the joy stick then executing the HJ command at the desired position. The position counters will then be loaded with zero.

KM HOME AND KILL

The KM command will find home and stop generating pulses immediately, i.e. no deceleration ramp will be generated. The position counter is not affected.

KR HOME REVERSE AND KILL

The KR command will find home in reverse and stop generating pulses immediately, i.e. no deceleration ramp will be generated. The position counter is not affected.

LM SEEK LIMIT

The LM command will cause the control to travel in the positive direction at the predefined velocity, until the limit switch is encountered, then stop generating pulses immediately. This command is useful in some home sequences to get positioned on the desired side of the home switch.

LR SEEK LIMIT IN REVERSE

The LR command will cause the control to travel in the negative direction until the limit switch is encountered, then stop generating pulses immediately.

MOVE SYNCHRONIZATION COMMANDS

These commands allow the synchronization of moves with external events or multiple axes sequences.

ID INTERRUPT DONE

The ID command will return the done flag ("!") to the host when the ID command is executed.

II INTERRUPT INDEPENDENT

The II command allows each axis to return the done flag to the host when it completes its move independent of the status of the other axes.

IN# INTERRUPT NEARLY DONE

The IN command will return the done flag to the host when the move is nearly complete. The parameter specifies the number of counts left in the move when the interrupt request is generated.

IC INTERRUPT CLEAR

The IC command will clear the done and error flags.

CA CLEAR AXIS DONE FLAG

The CA command operates like the IC command, except it clears the done flag of the addressed axis only.

WA WAIT FOR AXES

The WA command, only valid in the AA mode, allows a command to wait until all moves on all axes are finished before it executes.

WQ WAIT FOR QUEUE TO EMPTY

The WQ command is a special command that stops the board from processing any new command until the queue for the current axis mode is empty.

SW# SYNC WAIT

The SW command can be used to synchronize to external events by commanding the SRX to wait for the input line to go false.

WT# WAIT TIME

The WT command will wait for the specified number of milliseconds before proceeding with the next command.

SYSTEM STATUS REQUEST COMMANDS

These commands allow the host to request the status of various move parameters including the status of limit and home switches.

WY WHO ARE YOU

The WY command returns the model and firmware revision of the board or system being addressed.

RP RETURN POSITION

The RP command requests the current position.

RQ RETURN QUEUE STATUS

The RQ command returns the number of entries available in the command queue.

RA RETURN AXIS INTERRUPT STATUS

The RA command returns the state of the limit and home switches, and the done and direction flags for the currently addressed axis. The done flag is reset.

RI RETURN INTERRUPT STATUS

The RI command returns the state of the limit and home switches, and the done and direction flags for all axes. The done flags are reset.

QA QUERY AXIS

The QA command returns the status of the single addressed axis like the RA command, except the status register and flags are not affected.

QI QUERY INTERRUPT STATUS

The QI command returns the status of all axes like the RI command, except the status register and flags are not affected.

RC REQUEST ACCELERATION

The RC command will return the current acceleration or deceleration of the current axis or axes.

RV REQUEST VELOCITY

The RV command will return the current velocity at which the axes are moving.

RU REPORT POSITION IN USER UNITS

The RU command returns the current positions in user units.

USER UNIT COMMANDS

The following commands allow specification of move parameters in user defined units. The OMS controls will automatically convert all move parameters to these units once they have been initialized.

UU# USER UNITS

The UU command converts all move velocities, distances, etc. to user specified units by multiplying by the parameter given in this command.

UF USER UNITS OFF

The UF command turns off user units and causes the SRX board to use its default units.

USER I/O COMMANDS

The following commands allow manipulation and testing of the user definable I/O.

AN AUXILIARY ON

The AN command sets the auxiliary output to the high level. The open collector driver is off allowing the output to be pulled high by a pull-up resistor. It may be used to change power level on driver modules so equipped or as a user specified output.

AF AUXILIARY OFF

The AF command sets the auxiliary output to the low level. The open collector driver is turned on causing the line to be near ground. It may be used to change power level on driver modules so equipped or as a user specified output.

PA# POWER AUTOMATIC

The PA command will perform an AN command at the beginning of each move and an AF command after the move. See AN and AF commands.

SE# SETTLING TIME

The SE command allows specification of a settling time, in milliseconds, to be used before the power is reduced, when using the PA mode.

BL# BIT LOW

The BL command sets the selected general purpose output bit on, i.e. logic low.

BH# BIT HIGH

The BH command sets the selected general purpose output bit off, i.e. logic high.

BX BIT REQUEST IN HEX

The BX command returns the state of the general purpose input bits in hex format.

RB RETURN OUTPUT BITS

The RB command returns the direction of the general purpose I/O lines as they are currently defined in hex format.

MACRO CONTROL COMMANDS

MD# MACRO DEFINE

The MD command allows definition of a macro. The parameter specifies one of 20 available macros of 200 character length. The macro is terminated by an EOF (Control Z).

MX# MACRO EXECUTE

The MX command causes the specified macro to be executed. Macros can not be nested but may pass control to another macro.

TEACH MODE COMMANDS

JY JOY STICK MODE

The JY command causes the controller to enter the joy stick control mode where the velocity of the selected axes are under the control of the joy stick. The maximum velocity may be changed with the VL command prior to entering the JY command. The controller can be taught a position by pressing the right hand rocker switch on the joy stick module when at the desired position. The first depression of the button after entering the JY command will remember position 0, then position 1, etc. (up to 100 positions). The order may be modified by the JN command. The desired axes pair may be selected by switch B on the joy stick module. This switch may changed at any time allowing up to six axes to be positioned and then taught. The JY mode can be exited by entering a KL or SA command from the host computer or terminal.

JZ JOY STICK ZERO

The JZ command allows the zero velocity of the joy stick to be set. While in the JY mode, release the joy stick allowing it to go to neutral center position then enter JZ. This zero setting will be remembered as the electrical zero until the JZ command is entered again or an RR command is executed.

JD# JOY STICK DEADBAND

The JD command allows the user to set the desired deadband for the joy stick. The allowed parameter range is 0 through 10 counts.

JN# JOY STICK POSITION NUMBER

The JN command allows the sequence of positions being taught to be modified. The parameter supplied is the next position to be taught.

JT JOY STICK TEACH MODE

The JT command performs like the JY command while switch A is on. The serial port is not active but PLC commands may be entered through the parallel port from a PLC or switch register. The commands allowed through the parallel port are as follows.

PLC TEACH MODE COMMANDS	
COMMAND	**FUNCTION**
0-99	Execute JN Command
100-127	Not Valid

JR JOY STICK RUN MODE

The JR command allows the positions to be commanded through the parallel port. This command is active while switch A is off. The commands allowed through the parallel port are as follows.

PLC RUN MODE COMMANDS	
COMMAND	**FUNCTION**
0-99	Execute PT Command
100-119	Execute MX Command
120	Execute HA Command
121	Execute HJ Command
122-126	Not Valid
127	Execute KL Command

PT# MOVE TO POSITION TAUGHT

The PT command causes the stage to move to the specified position previously taught.

PE#,# POSITION ENTER OR EDIT

The PE command allows positions to be entered or modified without moving the stage. The first parameter is the position number followed by the desired positions. Entering a comma as a place holder with no position parameter will leave that axis unchanged allowing only the desired axes to be modified at any position.

RT# REQUEST TEACH POSITIONS

The RT command returns the position which has been previously taught. The parameter must be the position number from 0 to 99.

CONSTANT VELOCITY CONTOURING COMMANDS

The contouring command set allows the building of a command sequence which can later be executed at constant velocity for machine tool and other similar applications.

AF#,# AUXILIARY OFF

The AF command turns off any combination of auxiliary ports, when encountered in the command stream, allowing control of other peripherals such as a laser beam for machining.

AN#,# AUXILIARY ON

The AN commands turns on any combination of auxiliary output ports when encountered in the contouring command stream.

CD#,# CONTOUR DEFINE

The CD command allows entry of a contour definition which will start at the position specified. Any combination of axes may be used in the contour mode.

CE CONTOUR END

The CE command ends the definition of the contour sequence, i.e. terminate the CD mode or ramp to a stop and exit when the contour is executed.

CK CONTOUR END AND KILL

The CK command ends the definition of the contour sequence and stops pulse generation immediately without deceleration.

CR#,#,# CIRCULAR INTERPOLATION

The CR command causes the axes defined by the CD command to move in a circular pattern from the entry position. The parameters specify the center of the circle and distance to travel in radians. The CR command is only valid with contours of 2 axes.

CV# CONTOUR VELOCITY

The CV command allows the specification of the contouring velocity and clears any previously defined contour.

CX CONTOUR EXECUTE

The CX command causes the SRX controller to execute the previously defined contour sequence.

MT#,# MOVE TO

The MT command causes the axes defined by the CD command to move to the specified absolute position using linear interpolation at constant velocity.

RQ REQUEST QUEUE STATUS

The RQ command returns the number of entries available in the contouring queue.

ENCODER COMMANDS (OPTION E)

The following are encoder support commands for use with SRX option E controllers only.

The following are position maintenance control commands:

ER#,# ENCODER RATIO

The ER command allows specification of encoder ratio by entering encoder counts followed by motor counts, for position maintenance mode.

HV# HOLD VELOCITY

The HV command specifies maximum position hold correction velocity. This is the peak velocity which will be used while making position corrections.

HG# HOLD GAIN

The HG command specifies the position hold gain parameter. The position error is multiplied by this gain factor in determining the velocity during a position correction.

HD# HOLD DEADBAND

The HD command specifies deadband counts for position hold. The SRX will consider the control in position when the stage is within the specified parameter counts during position correction.

HF HOLD OFF

The HF command disables position hold, stall detection and tracking modes.

HN HOLD ON

The HN command enables position correction after a move.

IP INTERRUPT WHEN IN POSITION

The IP command operates like the ID command, except the interrupt is deferred until the stage is within the specified deadband.

The following commands control the slip or stall detection mode:

ES# ENCODER SLIP TOLERANCE

The ES command parameter specifies tolerance before slip or stall is flagged in the status register.

IS INTERRUPT ON SLIP

The IS command will enable interrupts to the host when the position error during a move exceeds the parameter specified by an ES command. A "$" will be returned to the host computer when slip occurs.

The following command controls the tracking mode of the controls:

RL RETURN SLIP STATUS

The RL command returns the slip detection status of each axis. An S is returned if slip has occurred for that axis, or else an N is returned.

The following command controls the tracking mode of the controls:

ET ENCODER TRACKING

The ET command turns on the encoder tracking mode. The axis will track its encoder input, thus allowing one axis to follow the activity of another or a thumbwheel for manual positioning or the movement of another device that produces a signal compatible to the encoder inputs.

The following commands control the home sequence when used with an encoder:

HE HOME ENCODER

The HE command enables the encoder index mode, i.e. home is defined as the logical AND of the encoder index, the external home enable and the encoder quadrant.

The following would be input from the host computer:

```
AA
CV1000
CD,,0,0;
AN0;
CR0,5000,3.1415926
CR0,0,6.2831853
AF0;
MT-10,10000
CE
MT,,-1000,0; GO
CX
```

In a move requiring a staircase of velocity with the breakpoints of velocity at absolute positions, the following commands would be used:

- Specify the X axis.
- Set the acceleration rate.
- Define a move in the positive direction.
- Accelerate to 30,000 pulses per second and move to position 10,000.
- Without stopping, accelerate to 50,000 pulses per second and move to absolute position 20,000.
- Without stopping, decelerate to 10,000 pulses per second and move to absolute position 30,000.
- Decelerate to a stop at position 35,000.

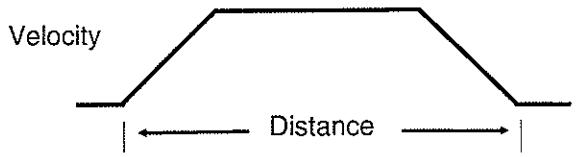
The following would be input from the host computer:

```
AX
AC100000
MP
MV10000,30000
MV20000,50000
MV30000,10000
SP35000
```

OPERATING PRINCIPLE

The SRX family of motion controllers provide acceleration to an optimum speed followed by a constant velocity and a controlled deceleration to a stop. The actual position can be simultaneously monitored by use of the encoder feedback option. The velocity profile is achieved by calculating the optimum velocity 1024 times each second providing a very smooth acceleration curve. The calculation is used to control a variable frequency pulse train which is derived from a crystal oscillator thus providing very accurate pulse rates. Linear as well as parabolic and cosine velocity ramps can be generated to fit a variety of system requirements. The following figure shows a typical acceleration and constant velocity followed by the deceleration using a linear velocity profile.

Moves may be chained together to provide a more complex pattern. An internal queue can store up to 200 command parameters in an input queue for each axis as well as 124 characters in an input character buffer. This allows a complicated move sequence to be performed without host intervention. A flag can be passed to the host on the completion of a sequence or at any intermediate point in the command stream.



Linear Velocity Profile

SPECIFICATIONS

Velocity

0 to 522,000 pulses per second simultaneous on each axis

Acceleration

0 to 8,000,000 pulses per second per second

Position range

134,000,000 pulses (±67,000,000)

Accuracy

Position accuracy and repeatability ±0 counts for point to point moves

Velocity accuracy ±0.01% of peak velocity in jog mode

Environmental

Operating temperature range 0 to 50 degrees centigrade

Storage temperature range -20 to 85 degrees centigrade

Humidity 0 to 90% non-condensing

Power

+5 volts at 1.88 amps typical (RS232 transceiver voltages developed on board)

Dimensions

Standard eurocard 9.187 x 8.66 x 0.75 inches high

Communication parameters

The valid baud rates are 19,200, 9600, 4800, 2400, 1200, 600 and 300 baud, 8 data bits, no parity and one stop bit. Uses DTR/CTS modem line handshake.

RS232 interface

Meets all RS232 signal specifications and definitions.

RS422 interface

Meets all RS422 signal specifications and definitions (mapped to RS232 style connector).

Limit switch inputs

TTL input levels with on-board 2.2K pull up resistor, requires only external switch closure to ground or TTL level input signal. Input sense (low or high true) selectable by on-board jumper for each axis.

Home switch inputs

TTL input levels with on-board 2.2K pull up resistor, requires only external switch closure to ground

or TTL low level input signal. Input sense (low or high true) selectable under software control for each axis.

User definable I/O

32 bits of user definable I/O. TTL input levels with on board 2.2K pull up resistors, requires only external switch closure to ground or TTL low level input signal. The auxiliary outputs are TTL open collector outputs (7406) which can be converted to TTL totem pole outputs (7404). The auxiliary outputs are fixed as outputs. 24 bits are configurable as inputs or outputs. The outputs are TTL nand gates (7400) and the inputs are TTL nor gates (7402). The output devices may be replaced with (7438) open collector devices when required. Factory default is 16 inputs and 8 outputs. These are jumper selectable with a change in IC type which is socketed.

Step pulse output

Pulse width 50% duty cycle. Open collector TTL level signal.

Direction output

Same as step pulse output

Joy stick interface

A joy stick interface is built into the SRX. It is compatible with 2000 ohm resistive joy sticks. Three control switches are also provided for selecting the axes pair, run/teach mode and a teach switch.

PLC INTERFACE CONNECTIONS		
PIN	**I/O BIT**	**FUNCTION**
A25	0	1
A26	1	2
A27	2	4
A28	3	8
A29	4	16
A30	5	32
A31	6	64
A32	7	Strobe
C32	23	Busy

In the PLC mode, commands may be placed on I/O bits 0 through 6 and strobed on bit 7. Bit 23 becomes true indicating a busy condition until the move or macro execution is complete. The commands are as described under the JR command. The remaining I/O lines are available for other user definable applications.

JY02-S SWITCH DEFINITION	
SWITCH	**FUNCTION**
A	Teach/Run Mode Selector
B	Axes Pair Selector
C	Teach Position (momentary)

JOY STICK INTERFACE CONNECTIONS	
PIN	**FUNCTION**
1	Unused
2	Unused
3	Unused
4	Unused
5	Unused
6	Unused
7	Second Axis Analog Input
8	First Axis Analog Input
9	Ground
10	Ground
11	Switch Input 1
12	Switch Input 2
13	Switch Input 3
14	Switch Input 4
15	+5 Volts

CONSRX CONNECTOR KIT

The CONSRX connector kit is available and supplied at no extra charge on orders delivered at quantity 1-4 pricing. It consists of the mating connector, hood and crimp style pins to make connections from the SRX to driver modules, limit and home switches, etc.

The mating connector is an Amp, Inc. part #532509-1 with a 532508-1 hood and strain relief. Crimp tabs are used, Amp, Inc. part #530151-6.

CONNECTOR PIN LIST

Pins are shown in their actual relative position as viewed looking into the connector.

NON-ENCODER VERSIONS			
PIN	ROW C FUNCTION	ROW B FUNCTION	ROW A FUNCTION
1	Ground	NC	+5 Volts
2	X Aux Output	X Direction	X Step Output
3	X Home Input	X Positive Limit	X Neg Limit
4	Y Aux Output	Y Direction	Y Step Output
5	Y Home Input	Y Positive Limit	Y Neg Limit
6	Ground	NC	+5 Volts
7	Z Aux Output	Z Direction	Z Step Output
8	Z Home Input	Z Positive Limit	Z Neg Limit
9	T Aux Output	T Direction	T Step Output
10	T Home Input	T Positive Limit	T Neg Limit
11	NC	NC	NC
12	Ground	NC	+5 Volts
13	U Aux Output	U Direction	U Step Output
14	U Home Input	U Positive Limit	U Neg Limit
15	V Aux Output	V Direction	V Step Output
16	V Home Input	V Positive Limit	V Neg Limit
17	Ground	NC	+5 Volts
18	R Aux Output	R Direction	R Step Output
19	R Home Input	R Positive Limit	R Neg Limit
20	S Aux Output	S Direction	S Step Output
21	S Home Input	S Positive Limit	S Neg Limit
22	NC	NC	NC
23	NC	NC	NC
24	Ground	NC	+5 Volts
25	User I/O 16	User I/O 8	User I/O 0
26	User I/O 17	User I/O 9	User I/O 1
27	User I/O 18	User I/O 10	User I/O 2
28	User I/O 19	User I/O 11	User I/O 3
29	User I/O 20	User I/O 12	User I/O 4
30	User I/O 21	User I/O 13	User I/O 5
31	User I/O 22	User I/O 14	User I/O 6
32	User I/O 23	User I/O 15	User I/O 7

ENCODER VERSIONS			
PIN	ROW C FUNCTION	ROW B FUNCTION	ROW A FUNCTION
1	Ground	NC	+5 Volts
2	X Aux Output	X Direction	X Step Output
3	X Home Input	X Positive Limit	X Neg Limit
4	Y Aux Output	Y Direction	Y Step Output
5	Y Home Input	Y Positive Limit	Y Neg Limit
6	Ground	NC	+5 Volts
7	Z Aux Output	Z Direction	Z Step Output
8	Z Home Input	Z Positive Limit	Z Neg Limit
9	T Aux Output	T Direction	T Step Output
10	T Home Input	T Positive Limit	T Neg Limit
11	NC	NC	NC
12	Ground	NC	+5 Volts
13	X Phase A-	X Phase A+	X Index+
14	X Phase B-	X Phase B+	X Index-
15	Y Phase A-	Y Phase A+	Y Index+
16	Y Phase B-	Y Phase B+	Y Index-
17	Ground	NC	+5 Volts
18	Z Phase A-	Z Phase A+	Z Index+
19	Z Phase B-	Z Phase B+	Z Index-
20	T Phase A-	T Phase A+	T Index+
21	T Phase B-	T Phase B+	T Index-
22	NC	NC	NC
23	NC	NC	NC
24	Ground	NC	+5 Volts
25	User I/O 16	User I/O 8	User I/O 0
26	User I/O 17	User I/O 9	User I/O 1
27	User I/O 18	User I/O 10	User I/O 2
28	User I/O 19	User I/O 11	User I/O 3
29	User I/O 20	User I/O 12	User I/O 4
30	User I/O 21	User I/O 13	User I/O 5
31	User I/O 22	User I/O 14	User I/O 6
32	User I/O 23	User I/O 15	User I/O 7

ORDERING INFORMATION

MODEL	AXES OF CONTROL	AXES OF FEEDBACK
SRX-2	2	
SRX-4	4	
SRX-6	6	
SRX-8	8	
SRX-2E	2	2
SRX-4E	4	4
SRX-6E	6	2
CONSRX		
JY02-S		

Oregon Micro Systems, Inc.
A Pro-Dex Company
1800 NW 169th Place • Suite C100
Beaverton, Oregon 97006
(503) 629-8081
Fax (503) 629-0688
Email: sales@OMSmotion.com
Web Site: http://www.OMSmotion.com

Part Number: E4DS-SRX-A
Revision A

INDEX

!

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W