SAX/DAX
USERS MANUAL

Congratulations!

............on your purchase of a SAX (single axis) and/or DAX (two axis) Stepper Motor Control System from ADVANCED MICRO SYSTEMS, INC. These units are designed to provide years of reliable, accurate and cost-effective motion control. As with all AMS products, the SAX/DAX are backed by nearly two decades of manufacturing excellence and a commitment to quality and support that guarantees your satisfaction.

This Technical Reference Guide will assist you in optimizing the performance of your system. It's purpose is to provide access to information that will facilitate a reliable and trouble-free installation. The Technical Reference Guide is organized into the following sections:

- Installation
- Serial Communication
- Auxiliary I.O.
- Operating Parameters,
- Programming Notes
- Program Commands
- Specifications
- Addendum

We recommend that each section be reviewed prior to installation.

In addition to the Technical Reference Guide, a systems integration software package (EASI) is supplied. This powerful programming tool is an easy to use, menu driven utility file with on-line help screens and available source code (Microsoft "C"). Used together, the Technical Reference Guide and the EASI diskette will enable you to quickly take advantage of the advanced programming features and system capabilities inherent in the system design.

Although the SAX/DAX and supporting documentation were designed to simplify the installation and on-going operation of your equipment, we recognize that the integration of motion control often requires answers to many complex issues. Please feel free to take advantage of our technical expertise in this area by calling one of our support personnel to discuss your application; 603-882-1447.

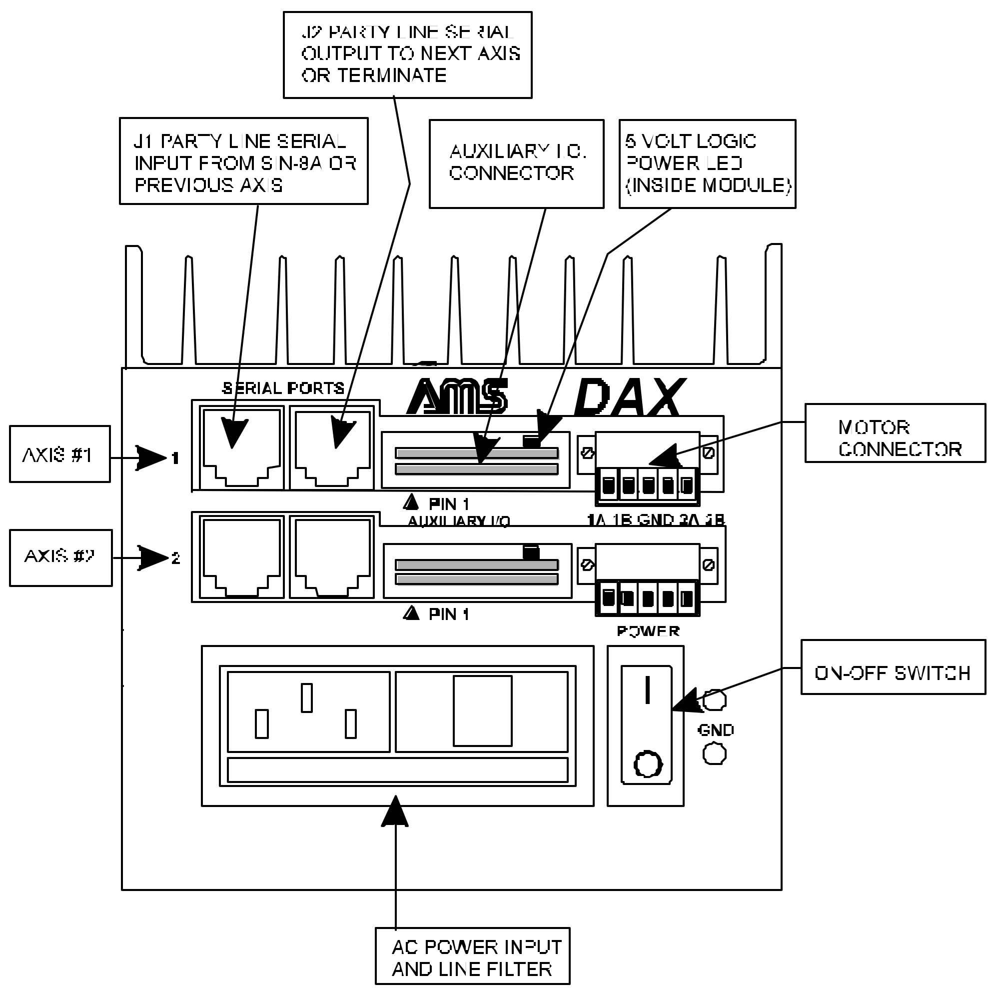
Thank You!
Your AMS Team

Required Hardware

Before installing the your stepper motor control system, inspect the contents of your package to ensure the following items are included:

Qty	Unit	Model #	Description
1	Axis	SAX/DAX	Step Motor Control System
1 1 1	Axis System System	SIN-7 or SIN-9 or SIN-8 or SIN-10	RS-232 serial adapter, 25 pin/9 pin (SAX) RS232/RS422 serial adapter and cable (DAX) Intelligent serial adapter and cable (SAX/DAX)
1	Axis	BLC-38	7 pin home/limit mating connector
1	Model	BLC-40	AC power cord
1	Axis	BLC-43	5 pin mating motor connector
1	Added axis	BLC-51-3	Interconnect cable, Cat5 (3 ft.)
1	System	TERM-2	Terminator plug (included with serial adapters)

Front Panel Description (DAX)



Hardware Set-Up

Prepare for operation by referring to the illustration on the preceeding page and following these simple instructions:

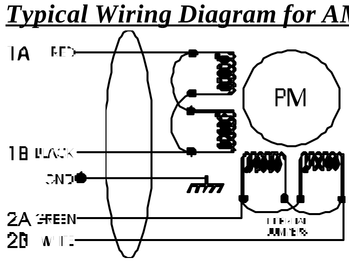
A. Attach the 25 Pin "D" connector end of the serial adapter to either COM1 or COM2 of your PC/AT compatible computer. If using a SIN-8 make sure that the RED switch on the side of the connector is in the single (S) line mode.

B. Connect the other end of the cable assembly (looks like a telephone connector), to the mating connector J1 "Party Line Serial Input", on the front panel..

C. Install a terminator plug (TERM-1) into J2 "Party Line Serial Output" of the last axis. The controller will work without a terminator plug, however reliability may be compromised, based on the environment (noise) and length of interconnect cables.

D. Connect a motor to the motor connector. Refer to the Addendum: "About Step Motor Current" for more information.

Typical Wiring Diagram for AM23 and AM34 Series Motors



Standard winding configurations are in parallel. Series windings are also available but require an internal jumper modification (made at the factory) and should be specified at time of order.

Note: NEVER connect or disconnect the motor when the power is "ON".

E. With the Power Switch off (down), connect the A.C. power cord to the power entry module on the front panel and to a correct A.C. power source (120V 50/60Hz or 240V 50/60Hz).

F. Turn the Power Switch on. The following actions should occur:
1. The fan on the rear panel should start rotating.
2. The 5 volt logic power indicator LED will illuminate (can be seen inside module, behind auxiliary I.O. connector).

Short Circuit Protection

The short-circuit detection circuitry will protect against inadvertent shorting of the outputs, and faulty loads; and the fuse will protect against a failed output driver. The short-circuit detection circuitry disables the output drivers. It is electronically latched and will trip at about 10 Amps of load current within about 2µsec. The fault condition can be cleared by either cycling the power to the controller, or by disabling, and then re-enabling the driver card (See "E" command).

Note: The Fault signal is provided to either the controller or the user, so programs may continue to run under this condition (without motor activity).

Software Set-Up (EASI Diskette)

EASI is a DOS program tailored to operate with AMS motion control produts. It is not intended to operate as an end user application program, but rather to allow familarization and evaluation of the products performance.

Almost any IBM PC/AT compatible computer will work with EASI. Although EASI will operate from a CD or floppy disk drive, copying the contents to a subdirectory (call it "easi") will increase the speed and performance.

Note: It is recommended that you review the "Getting Started" instructions on the EASI diskette by browsing the "README" file.

A. Install the diskette and type: EASI<CR> at the prompt. At the opening screen enter "n" if you are operating from a monchrome terminal or hit the ENTER key for color.

B. Use the arrow keys to select COM1 or COM2. COM1 is the default setting. Follow the same procedure to select the correct Baud rate. 9600 BPS is the default setting.

Sign-On

C. Select the "DUMB TERMINAL" mode. A blank window will appear.

D. Strike the SPACE BAR key. The controller should sign on with the software version number Vx.xx. If not, enter a (^C) (reset) and strike the SPACE BAR key again.

Striking the ENTER <CR> key should result in an echo of "# " characters, further indicating communication is established.

Name Axis RS-422 "Party Line" Mode

Each axis should be assigned a unique "name" for proper operation when used in Party-Line mode.

E. Reset the controller (^C), then type a single, valid (upper or lower case) name character:

Note: Make sure that the RED switch on the SIN-8 connector is in the single (S) line mode.

Recommended Names: Upper case A through Z Lower case a through z		Non-valid Names	
ASCII	HEX	ASCII	HEX
[	5B	^C	03
\	5C	CR	0D
]	5D	LF	0A
^	5E	@	40
-	5F		
‘	60		

F. Follow the name with a SPACE BAR. The sign-on message will appear.

G. Verify the Name by entering the "X" command <CR>.
The last item of the first line should contain the "Name" character.

H. Enter the Save command (S) <CR>. The axis Name is now stored in non-volatile memory.

The unit is ready to operate in the current single axis mode or be switched over to Party Line mode. It is suggested that the operator use single mode first to become familiar with command input. The single axis mode can be used with any "dumb" terminal device and is not dependant on using the AMS software. For multi-axis operation refer to "Party Line Startup" in the Communication section of this manual.

Note, single axis, RS-232 mode operation does not require the name axis procedure.

Examine Command

The Examine command (X)<CR>. will display a set of parameter values that were last stored in non-volatile memory. These parameters may be modified using the appropiate commands, then stored in non-volatile memory as the new "defaults."

Default Parameter	Value
C	10
K	5
I	400
V	5016

Where:
C= Hold Current %
K= Ramp Slope
I= Initial Velocity
V= Slew Velocity

The values shown assumes there are no input connections or special modes such as inverted limit switches.

Simple Program Examples:
1. Issue the command: “R -1000”<cr>. The motor should move.
2. Issue the “Z”<cr> command. The position (-1000) should be displayed.

Some Rules

1. The command line may be edited using back-space as characters are typed.
2. The line may be canceled using <ESC>.
3. The command line is limited to 12 characters.
4. Only one command may be entered per line.
5. A space is optional between the command and first number
6. A space or comma must be used to separate two parameter commands.

Entering Programs

The above examples were samples of immediate commands. The folowing is a sample of the sequences that are stored in non-volatile memory. Note, when programing the sequence is immediately written to non-volatile memory, without any additional action required to save it.

Address	Enter	Remark
	P0<CR>	Place in Program mode. Insert instructions at location 00.
0	O0<CR>	Set Origin to zero.
1	R10000<CR>	Move 10,000 steps in the “+” direction, relative to Origin.
6	W0<CR>	Wait until complete.
9	R-10000<CR>	Move 10,000 steps in the “-” direction, relative to Origin.
14	W0<CR>	Wait until complete.
17	J1 3<CR>	Jump to address 1, 4 times.
21	R500<CR>	Move 500 steps in the “+” direction, relative to Origin.
26	P0<CR>	End Program.

Now list the stored program:

	Enter	Remark
	Q<CR>	Query command.

Note: An upper case “Q” displays one line at a time. A lower case “q” will display up to 25 lines at a time.

Verify the Program

The controller will respond with:

0	O	
1	R	10000
6	W	0
9	R	-10000
14	W	0
17	J	1 3
21	R	500
26		

Execute the Program

Enter	Remark
G0 1<CR>	Programs start executing at location zero. If the Trace option is on, it will display each instruction, prior to execution.

Note: The program can be terminated at any time by hitting the ESCape key.

Edit Program

Example: It is desired to change instruction number 21 from 500 steps to 5,000 steps:

Enter	Remark
P21<CR>	Edit instruction 21.
R5000	Move 5,000 steps in the "+" direction, relative to Origin.
"ESCape"	Terminates Edit mode.

Note: Caution should be exercised when making Program Edits in dumb terminal mode due to variations in Command byte length that may effect subsequent command address locations and possible corruption of non-volatile memory storage. It is recommended that application programs be developed using the menu driven program (Party Line selection) in the EASI diskette which includes a sophisticated Editor and Compiler.

Serial Command, Single Mode

The words “host” and “terminal” are used interchangeably and refer to any source of commonly available standard RS-422 interface including; terminals, printer ports and UARTs (COM ports). A character is a stream or packet of 8 consecutive bits (1s and 0s) with a defined sequence. Each unique character represents a letter, number, punctuation or unreadable control character as defined by an international set of standards called ASCII. The speed that characters are sent or received is defined by the Baud rate. The serial communication link refers to any two wire, full or half duplex communication link.

Command lines consist of an ASCII character followed by 0, 1 or 2 decimal ASCII numbers depending on command requirements. The User may edit the line prior to entry by using either the BACKSPACE or DELete key. The command line may be up to 12 characters long, including spaces. Spaces are optional between the command character and first number. Motion commands with 2 numbers require at least one space between numbers. Motion commands can either be upper or lower case.

In the Command mode, the command is executed upon receipt of a Carriage Return (Single mode) or ^J, ^Enter (Party Line mode). The controller will respond with a Carriage Return and Line Feed upon acceptance of the command.

RS-422/485 Party Line Protocol

Communication options include RS-422/485 Party Line protocol, recommended for multi-axis applications. In this mode each receiver monitors the host and responds when receiving a matching Name character. Communication to each individual axis in Single mode is performed to assign the unique Name (s) for each slave axis.

The Single mode provides User friendly one axis communication for setup and debug functions. Setup usually involves optimizing operational parameters, writing and storing programs and assigning a unique Name for Party Line operation.

Party Line Hardware

“Party Line” products have a 4 wire RS-422/485 interface. Differential line drivers and receivers are used to provide reliable communication in noisy environments. This design allows a single Master (or Host) computer and up to 32 Slave controllers. The hybrid design retains all the desirable characteristics of both EIA RS-422 and RS-485 specifications. In general, the hardware implementation follows the extended RS-485 standard with higher voltage and receiver capacity. Rather than half duplex protocol however, a full duplex Party Line communication is provided.

Slaves

Each Slave unit contains the following:

1. RS-485 line receiver; always active.
2. RS-485 tri-state line driver; activated on receipt of “address received.”

The line driver is always enabled when Single mode operation is selected.

Master

The Master interface, consists of:

1. RS-485 line receiver; always active
2. RS-485 line driver; always active

A Master interface, using the SIN-8 (RS-232 to RS-422) modular adapter, also contains a switch to select Single or Party Line mode.

Note: The RS-485/422 is rated for a maximum cable length of 4,000 feet. It is recommended that a second terminator (120 ohms) be used if runs exceed 15 feet or if operated in an electrically noisy environment. DO NOT bundle signal wires and motor wires together. The high current and frequencies generated by chopper drivers will couple, even if shielded wire is used (unless 100% shielding can be guaranteed). Avoid proximity to relays, motors and other RFI sources.

Moving

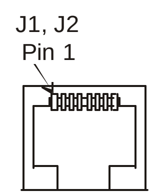
The Moving signal is an open collector buffered output on all AMS motion control products. The SIN-8 converts this signal to RS-232 level and supplies it to Pin 8 “Carrier Detect” of the 25 Pin “D” connector.

SIN-8 Adapter Module

Once the unique Name has been assigned and stored in the non-volatile memory, multiple axis may be connected in parallel for Party Line mode operation. A switch on the SIN-8 (RS-232 to RS-422 adapter module) allows convenient selection between modes.

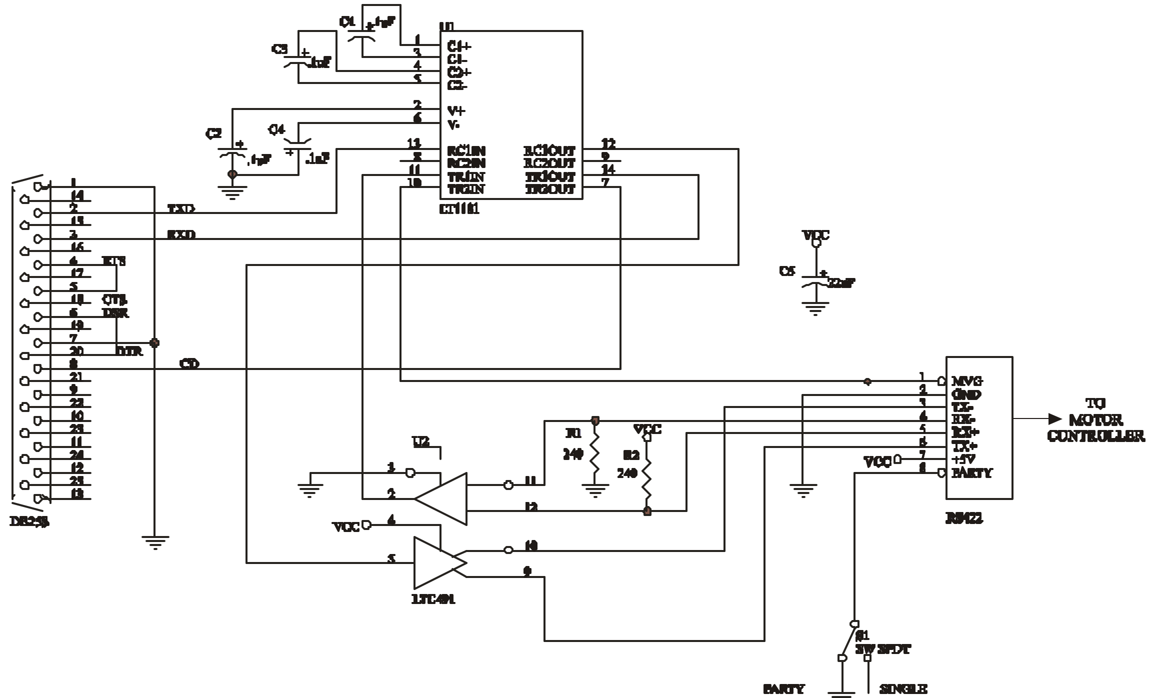
Single mode communication enables terminals or computers with serial communication capabilities to interface with the built-in command line editor. Party Line protocol requires the correct input character sequence to address an axis. It is recommended that a PC or other computer be used to simplify programming.

Serial Interface Connector (RJ45) on SIN-8			Serial Interface Connector (J1) on SAX/DAX		
Pin	Signal	Type	Pin	Signal	Type
1	Moving	Output	1	Moving	Output
2	GND	Logic	2	GND	Logic
3	TXD-	Output	3	RXD-	Input
4	RXD-	Input	4	TXD-	Output
5	RXD+	Input	5	TXD+	Output
6	TXD+	Output	6	RXD+	Input
7	+5v	Logic	7	+5v	Logic
8	Party Enable	Input	8	Party Enable	Input



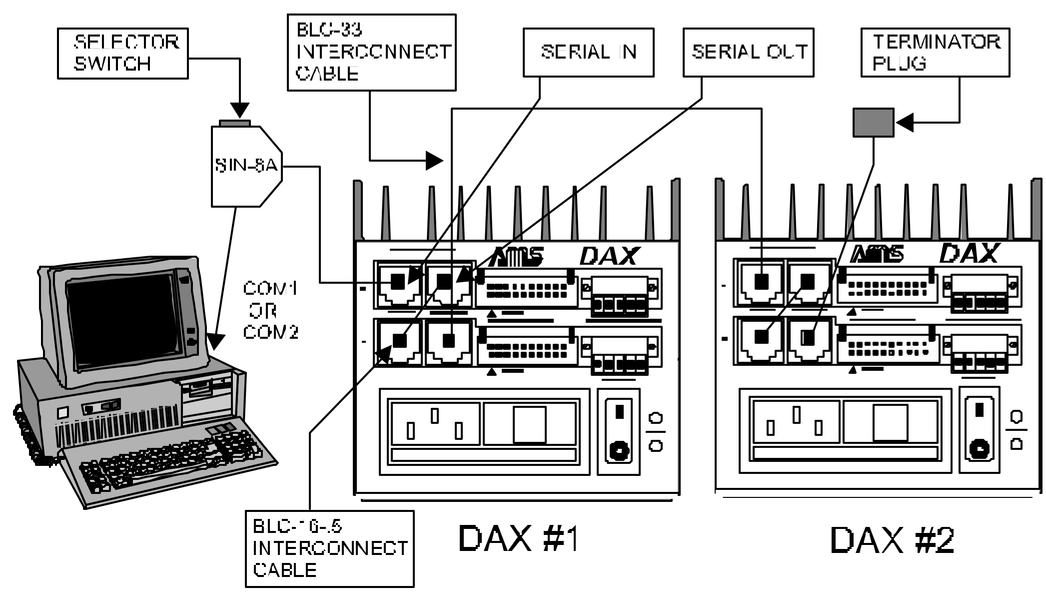
Party Line operation provides for parallel communication of up to 32 axis from a single serial communication port.

SIN-8 Schematic



Party Line Startup

After assignment of a unique “Name(s)” connect the axis (1 to 32) as shown. ***Move the selector switch on the SIN-8 or SIN-10 to Party Line (P) mode.*** When starting in Party Line the Name is read from non-volatile memory and then skips any sign-on procedure. At this point the sleep condition (serial outputs tri-stated) exists until a Name match occurs.



Serial Protocol

Each Slave unit residing on the Party Line acts as a listener, waiting for it's personal address (name) character. Once the proper name is received, the Slave enables it's RS-485 driver onto the TX bus. The activated Slave then echoes the start character and receives and echos the remainder of the command string until the terminator is received. The terminator must be a "line feed" (lf).

Once the line-feed character has been received, another axis may be commanded. In order for any axis to recognize a name, the name must be preceeded with a line-feed (the previous terminator will do).

The echoed characters provide reliable handshaking and a means of receiving data from the Slave, i.e., a "Z" position register command.

Party Line communication using a dumb terminal might go like this (assume 2 axis with address "X" and "Y"):

Master (Host PC): <lf> X+1000 <lf> Y-500 <lf> XZ <lf>
Slave (Axis): X+1000 <lf> Y-500 <lf> XZ1000 <lf>

ASCII Baud Rates

Baud rates for communication is fixed at 9600 BPS. Consult AMS if an alternate communication rate is required. Host/terminal settings are:

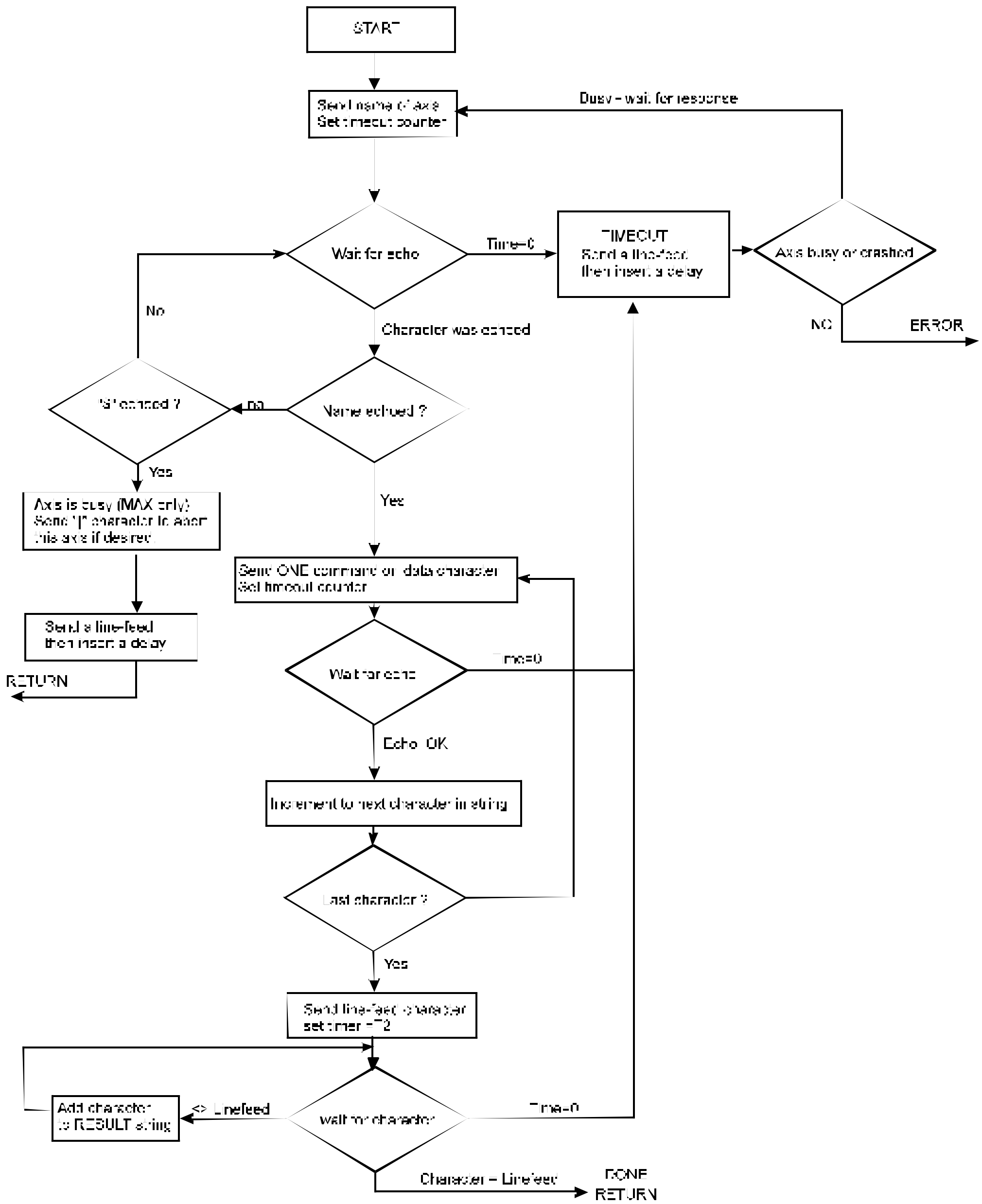
Start Bits	Data Length	Stop Bits	Parity
1	8	1	None

Program Considerations

The SAX/DAX incorporate a buffered UART input. Because motion control is of the highest priority, processing of received information may be delayed if commands are sent while stepping at very fast rates. This condition may only occur at internal/external step rates exceeding 10,000 steps per second. In serial applications where commands are sent while motion is active, the User should monitor echoed data to avoid UART overrun.

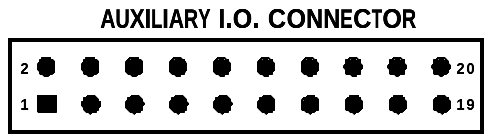
Note: The host computer must verify the character echo as each character is sent. Inserting a delay between character will inevitably fail and slow communications. Your application software must be capable of single character (as opposed to string) transmision without reception loss at 9600 Baud.

Party Line Communication Flow Chart



I.O. Connections

Twenty contact header connectors provide auxiliary inputs and outputs. Buffered signals use optical isolation or TTL buffers. It is best to visualize these signals as “digital” signals. When the signal is at an off-state, it will be high or +5 Vdc (up to 8 Vdc for Optically Isolated Inputs). When the signal is at an on-state, it will be at 0 Vdc.



Pin	Signal	Type	Typical Connection Drawing
1	Port 2	TTL Input	Fig. 1
2	+5V	Voltage Supply Output	
3	Port 4	TTL Output	Fig. 2
4	Home	Optically Isolated Input	Fig. 3
5	Moving	TTL Output	Fig. 2
6	Limit A	Optically Isolated Input	Fig. 3
7	Port 5	TTL Output	Fig. 2
8	Limit B	Optically Isolated Input	Fig. 3
9	Port 3	TTL Input	Fig. 1
10	Jog -	Schmitt-Trigger Input	Fig. 1
11	Port 1	TTL Input	Fig. 1
12	Jog +	Schmitt-Trigger Input	Fig. 1
13	Jog HS	Schmitt-Trigger Input	Fig. 1
14	Trip	TTL Output	Fig. 2
15	GND	Power Common Ground	
16	Soft Stop	TTL Input	Fig. 1
17	Go	Optically Isolated Input	Fig. 3
18	Port 4	TTL Input	Fig. 1
19	+5V	Voltage Supply Output	
20	Opto Supply	Opto-Isolator Power-in	

Inputs are TTL with 10k pull up resistors. Outputs are open collector TTL (7407) with 10k pull up resistors.

Applications requiring optical isolation may supply power (8 volts max) or jumper from +5V to Opto Supply. The optically isolated input’s are the Limits, Go, and Home. ***NEVER attempt to draw current from the OPTO+ pin.***

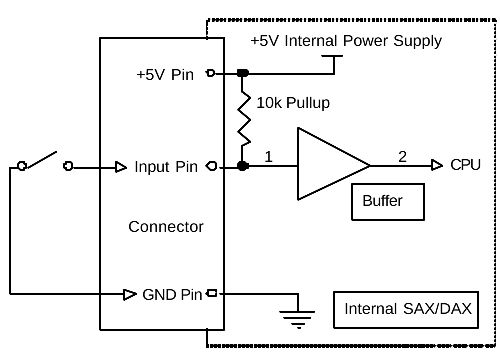


Figure 1

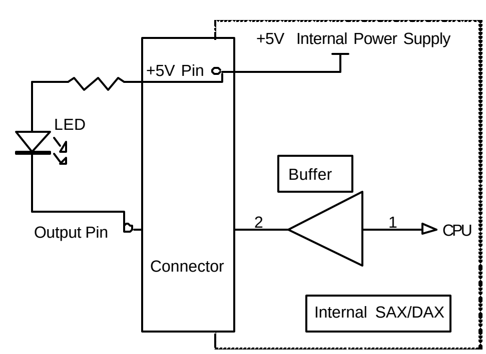


Figure 2

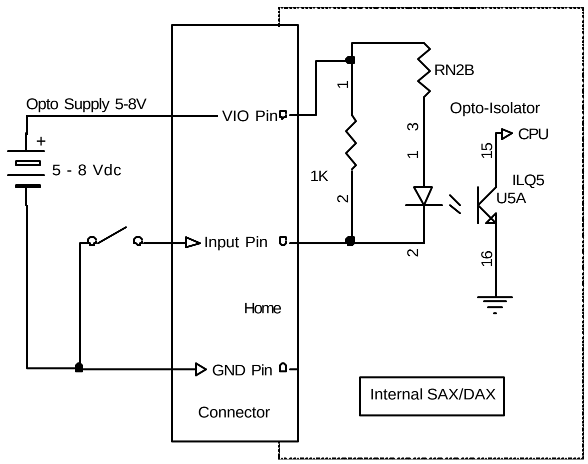


Figure 3

General Purpose Inputs

P1-4 IN, The General Purpose Inputs provide four user programmable inputs supporting conditional input looping with the Loop On Port, or "L" command; and conditional branching using the Vectored Go, or "G2048" command. Port 2-4 are bi-directional I/O in that the port can be used as an input if the output bit is turned off. Inputs status can be read using the Read Port command, "A129."

The Loop On Port command is useful for waiting for an input to become active before proceeding with a program. The vectored GO command is especially useful for selecting various programs with an external thumbwheel switch, or other logic devices, which can select various input, bit combinations. P4 is a bi-directional I/O in that it can serve as either an input, if the output is turned-off; or as an output if the input is not needed.

Dedicated Inputs

The dedicated inputs: LIMA, LIMB, HOME, JOG1, JOG2, JOGHS, GO and SS support motion specific functions and cannot be programmed for other purposes.

LIMA and LIMB, the Clockwise Limit, and the Counter-Clockwise Limit, respectively, are used as an over-travel input to prevent equipment damage or personnel hazard from traveling beyond the bounds of the work envelope through a programming, or operational error.

The HOME input is used to reference the axis to a specific location and can be an independent switch, or it can be connected to one of the limits. Home is only examined in conjunction with the Find Home, or "Fxxx" command.

JOG+ and JOG-, cause the motor to move, or jog clockwise, and counter-clockwise, respectively; both at a speed programmed by the Set Jog Speed command, "Bxx xx."

JOGHS causes the JOG1 or JOG2 inputs to operate in the high-speed mode, as programmed by the Set Jog Speed command, "Bxx xx."

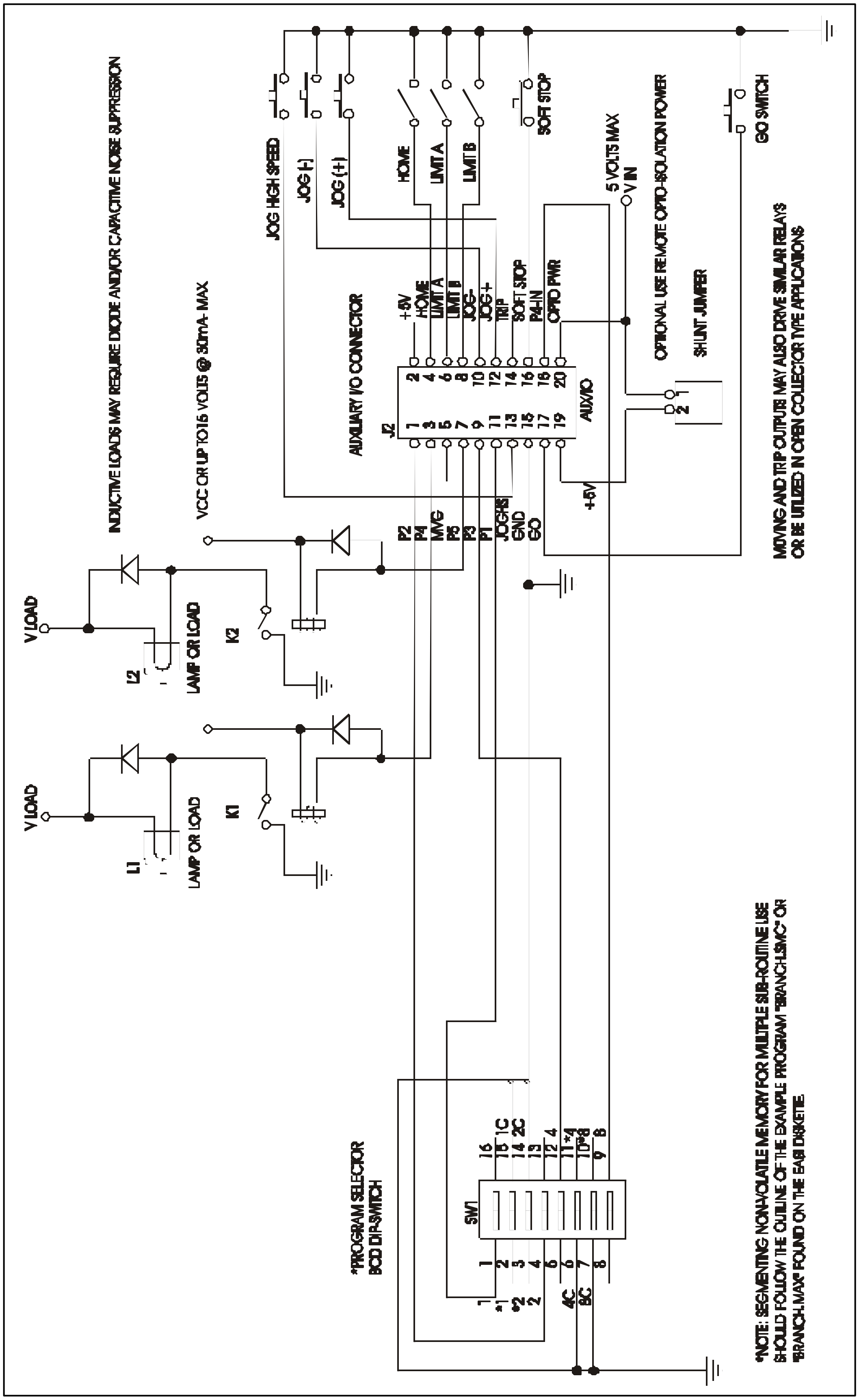
The GO input causes program execution to begin at program address location 0. This function is useful for stand-alone operation, when a program is stored in memory.

The SS, or Soft Stop input. If executing a program, activating this input causes program execution to stop. If executing a move, activating this input causes the move to ramp to a stop. Since activity termination is not immediate, and because this functions under software control, **this input should not be used in place of an emergency stop.**

General Purpose Outputs
The P4-5, Trip general-purpose outputs provide user programmable outputs. These outputs can be programmed using the Port Command (‘A xxx’). P4 is a bi-directional I.O., in that it can serve as either an input, if the output is turned-off; or as an output, if the input is not needed. P4 OUT, further reflects the state of P2-4 IN, if programmed as an input.

Dedicated Output
Moving is a dedicated output, which is active whenever the controller is stepping. This output is useful as a ‘Moving Status’ bit.

I.O. Schematic



Reset/Initialize

Automatic hardware reset occurs each time the power is turned on or a command “^C” is issued from serial communication. During reset all inputs and outputs will be at a high state. After hardware reset all parameters are initialized to factory set default values. The communication mode is established at 9600 BPS. Additionally, checks for Party Line operation are initiated.

Resident non-volatile memory is accessed, and the parameters most recently stored by the “S” command are downloaded, replacing the standard defaults in the working registers of the controller. The following block of parameters are stored and recovered as a set:

Parameter	Standard Defaults
Initial Velocity (I)	400 (steps per second)
Slew Velocity (V)	5016 (steps per second)
Divide Factor (D)	1
Ramp Slope (K)	5/5
Jog Speeds (B)	30, 200
Trip Point (T)	Off
Full/Half Step (H)	Full
Auto Power Down (E)	Yes
Hold/Run Current (Y)	10, 70
Limit Polarity	Low
Auto Position Readout (Z)	Off
Name (after reset)	Undefined

Note: Commands that modify these parameters use the working registers inside the controller. Actual non-volatile memory storage is initiated by the “Save” command. Once initialization is complete, Jog and Go inputs are active to allow jogging or a low pulse on the Go input to execute a program previously stored in non-volatile memory. A terminal or host is NOT required for these functions and may be initiated from the Auxiliary Input/Output connector.

Input Command Example

Command	Single Axis Mode Operation
+1000 (Carriage Return)	Step 1000 steps in + direction
+ 1000 (Carriage Return)	Same as #1
H 0 (Carriage Return)	Set fixed step resolution
H0 (Carriage Return)	Same as #3
H (Carriage Return)	Same as #3 (0 is used by default)
R -1000 (Carriage Return)	Move to position -1000

Commands such as Jump and Loop instructions are only valid when used in the Program then Execute mode.

The following can only be executed from programs stored in optional non-volatile memory:
J 0 5 (Carriage Return) Jump to location 0, 6 (n + 1) times.
J0 5 (Carriage Return) same as above.

Result Data

Some commands result in a numerical display. These consist of whole numbers that may have preceding spaces and are followed by a Carriage Return and Line Feed character. Negative numbers are preceded by the minus "-" sign.

Party Line Commands

During Party Line operation characters will NOT be echoed to the host until the proper "Name" (preceded by a ^J or ^Enter) is detected. All axis monitor concurrently the common TXD line from the host. Once the Name is received, the target axis will wake-up and start echoing as described above.

Instruction Execution

For each Motion command there are four cycles:

1. Entry
2. Execution
3. Result
4. Completion

All other commands have three cycles:

1. Entry
2. Execution
3. Result

In the idle state the controller continually tests for Jog, Go or Command input. The following describes each sequence operation that takes place on receipt of a command:

Cycle 1. Entry

The input commands and data are loaded via RS-232 or RS-422/485 interface. Command and data information is placed in a command line buffer as received. Editing is permitted in Single mode. ESCape aborts operation and returns to an idle state. A Carriage Return (^J or ^Enter for Party Line) terminates the Entry cycle and initiates execution.

Cycle 2. Execution

The command is processed. In the case of two consecutive action commands, execution will be delayed until any previous completion cycle has been completed.

Cycle 3. Result

The result cycle outputs any numerical result required by the command, i.e. the position.
The result type is signed numerical data, preceded by space padding and followed by a Carriage Return and Line Feed. If the result does NOT produce numeric data then the Carriage Return, Line Feed output indicates execution is complete.

Cycle 4. Completion

The completion phase is required for any Action command cycle. The following are Action commands:

Action Command	Completion Cycle
Go	Until last instruction is complete
Half/Full Step	Until previous action is complete
Wait	Until previous action is complete
Constant Speed	Until previous ramp is complete
Find Home	Until home is found
Relative Move	Until full index is complete
+Step Index	Until full index is complete
- Step Index	Until full index is complete

During the Completion cycle (except for Go) any non-action command, such as read position, may be executed. Another action command will be queued-up during the Completion cycle of a preceding Action command. The Execution and Result cycle of this Pending command is delayed until the Completion phase is finished. This interval is called the Pending Period. During this Pending Period the command accepted is the one character interrupt (abort) command, limit switches (J2 pins 6, 8) soft stop input (pin 16) and home switch (J2 pin 4).

External indication of Pending Period end, Execution and Result cycle of the pending instruction is the Carriage Return. The Go command is regarded as a command that has a continuous Pending (Instructions Queued) Period.

Interrupt Commands

Interrupt commands are single character commands that will interrupt the operation in process as follows:

Abort:

Any Action command may be terminated using the ESCape command.

Process	Resulting Action
Command line input	Clear input buffer
Program mode	Exit without inserting "End"
Action command	Terminate all motion Hard Stop
Program execution	Terminate execution Hard Stop

Note: All process(es) are aborted upon ESCape.

Soft Stop “@”

The Soft Stop "@" can be either a command (Immediate mode) or a single character interrupt (Program mode). The Soft Stop operates only when motion resulting from action commands or instructions is taking place.

Soft Stop Interrupt

After velocity deceleration the process is terminated.

Process	Resulting Action
Pending period	Decelerate and cancel pending instruction.
Program execute	Decelerate then terminate execution.

During Pending periods that are a result of Multiple and Constant Velocity commands (inter-speed ramping) and deceleration will be delayed until the previous ramp-to-speed has been completed.

Jog Speeds, Homing

Jog input and home speed is a special case of the constant velocity command. Inter-speed ramping is used if the programmed jog speeds are above the initial velocity. Homing does NOT employ a deceleration ramp on reaching the home sensor.

Note: In any mode jogging and command reception are mutually exclusive. A command can NOT be loaded while jogging and jogging can NOT be performed until the last command is complete. A command starts with the reception of the first command character.

High Speed Considerations

Step rates are controlled with a high degree of accuracy. As a result, step control is given priority over other processes. At high step rates this will manifest itself as a slight latency. The execution time increases when high step rates are active during command cycles. An example might be reading positions while moving at a high speed. Usually this latency has little affect at step rates below 10,000 steps per second. At speeds approaching the maximum step rate the processing latency may have to be taken into account.

Related Items: Trip and Loop

The Trip point output (J2 pin 14) is activated on the exact step position specified. When running a program (from the GO command) several "fetches" from the non-volatile memory are required along with the service time. This latency may allow several motor steps to occur before the desired action takes place.

Loop on port may exhibit similar latency effects at high speeds. The port will require a longer "true" condition to be recognized. A faster method is to implement the "wait for port" condition using the "Go -Sub," (branch on port) condition.

Non-Volatile Memory

The SAX/DAX host a 2048 byte non-volatile memory for each axis. The non-volatile memory may be used to store User programs for future execution via the "Go" command. Any number of programs may coexist, limited only by the available memory space.

The following memory map indicates that address locations are segmented into 8 pages and are accessible through direct read/write commands or cleared using the appropriate "C" command:

Location	Description
(0-226)	User program or data storage
(256-2048)	User program or data storage
(227)	Configuration byte
(228)	Internal initial status byte. (Do NOT modify)
(229)	Divide factor (D)
(230,231)	Initial velocity low and high bytes
(232,233)	N/A
(234,235)	Slew speed (V) low and high bytes
(236,237)	N/A
(238)	Low speed jog value
(239)	High speed jog value
(240)	Ramp factor acceleration (K)
(241)	Ramp factor deceleration (K)
(242,243,244)	Trip
(245)	Hold current
(246)	Run current
(247)	Name for Party Line

A special memory "Scratch Pad" of 64 bytes is accessed in location 128 to 192 for use in the working registers of the controller.

The EEPROM has a finite life of approximately 400,000 “write” cycles. Care should be used when writing to non-volatile memory to exclude unnecessary write cycles. For example, the Restore command (“^C” from a terminal) will retrieve the parameters from the EEPROM without doing a write. If the Initialize command (“C 1”) was chosen, the first 256 bytes of EEPROM are written. Should you require a sequence of motions to be done without host attention, you may breakup the motions into subgroups rather than repeatedly programming the EEPROM. Use the Go from address command to execute the subgroups in the required sequence.

Use the Save command sparingly. Parameters are set so quickly that it is sufficient to just let the host download them. Changing parameters should NOT be done by writing directly to EEPROM. The unit will not know that it was changed and may initiate an overwrite. Use the commands available to set parameters. Unlike writing, reading is non-taxing on the EEPROM.

Instructions

In the Command mode, commands are normally executed as soon as they are entered. The use of non-volatile memory allows storage of a list of commands. These stored program(s) can be triggered at power-up for automatic or repetitive operations by initiating a command or by strobing the Go (input J2-17) to a logic low. When in the Program mode, the entered commands (now called instructions) are directed into the non-volatile memory. After leaving Program mode, the stored program(s) may be subsequently executed by entering the "G" (Go) command.

The following procedure assumes a standard (RS-232) serial interface using a common terminal:

The Program mode is initiated by entering "P aa" (Carriage Return). The desired start address "aa" is chosen by the User. Generally, address 0 is a good choice for the main program because a program located at address 0 can be started with a simple "G" (Carriage Return) or by strobing the "Go" (input J2 pin 17).

Once in the Program mode the current memory location is displayed on the terminal and instructions may be entered. As each instruction is entered, the location is displayed. All instructions have the same format as in the Command mode.

Terminating the Program mode is done by entering a "P." This will cause the end of program flag to be inserted and the controller will echo the pound (#) character. It will then return to the Command mode.

Several programs may co-exist in memory. Each program may be executed independently by issuing a "Go" command with the appropriate address. The length and quantity of programs may occupy the full 2k byte of memory space.

Note: The end of program indicator occupies one additional byte. A program sequence that will be "called" when a trip point is passed may be located at an address defined by the trip point.

Editing Programs

Existing program(s) may be modified at any time. The User can review the existing instructions by entering the “Q” command. This command produces a list of instructions along with their memory addresses. To edit an existing program enter “P”, along with the desired address, and proceed to enter the new instruction(s) as in the Program mode. The edit session may be terminated in two ways. If the edit results in a program that is longer then the previous program or if the User wishes to discard the old instructions (shorten program), enter “P” to terminate edit and cause an end of program marker to be inserted. If only one or several successive new instructions are to be altered, entering ESCape will terminate the edit. Any instructions outside of the edit area will NOT be altered.

Note: If any instructions are of different byte lengths than existed previously, the program could wind up with invalid instructions in the middle of the program. Keeping track of the byte count will avoid this condition. The User may insert redundant or “dummy” one byte instructions to fill the gap. If in doubt reenter the remaining portion of the program

Single Step

Instructions may be used with a "Single Step" attribute. Instruction/commands that have this attribute will wait until a low trigger level is input to port 1 (J2, pin 11). This feature is valuable in implementing a master/slave type system.

The "master" can generate trigger pulses to the slaves that are waiting. The trigger may be any source capable of generating pulses between 10 and 100 microseconds. Entering a period "." will also generate a "software" trigger.

The trigger is DC level sensitive and is equivalent to having a "LO$" command preceding a standard instruction. Consecutive instructions with this attribute will be executed as long as the port is held low, (for either a single input or previous set port (A1) command).

The "Single Step" attribute is implemented during terminal entry by terminating the instruction command line with a period ".", rather than a Carriage Return. The instructions maintain the attribute in both Program Storage and Execute Now modes. Instructions are listed (using the Query command) with a period "." immediately after the instruction letter.

Trip Function

A position sensitive trip point may be set. When moves are made in Command mode the Trip Port changes state each time the specified position is reached. During program execution a User program located at address 200 will be automatically "called' when the trip point is passed. The program (at 200) will execute even if movement resulting from the previous instruction is still active. Setting trip position to +0 disables the trip function. Setting T to -0 will define position 0 as the trip point.

The logic state of the trip output may be toggled by use of the "A 64" command. Reading the trip status by a host can be accomplished by using the "A 128" command and testing bit 6 (64 decimal). Some delay (milliseconds) will be experienced, as a result of the above mentioned "call", but the trip output will still occur at the exact position specified.

Program Command Description

Command	***Function***		***Type***		***NV Bytes***
	Mnemonic	***Data 1*** ***(Range)***	***Data 2*** ***(Range)***		***Result***

Where:

Command:
ASCII character (Keystroke).

Function:
Functional description of command.

Type:
D= Default (initial parameter setting), I= Immediate (direct execution), P= Program command; executable in stored program.

NVBytes:
Non-volatile memory byte requirements in program..

Mnemonic:
Single character prefix used in multi-axis protocol; (prefixed by axis "Name" assignment in Party Line mode).

Data/Range 1:
Affected parameters and valid numerical range of parameters.

Data/Range 2:
Same as Data Range 1 (as required).

Result:
Information returned as a result of command execution or examination.

Command	***Function***		***Type***	***NV Bytes***
ESC	Terminate Operation		Immediate	0
	Mnemonic	***Data 1***	***Data 2***	***Result***
	(Name) Esc. Character	None	None	Echo

ESC (Global Abort)

Global Abort terminates any active operation and forces the controller to revert to the idle state. Output drivers or ports are not effected. Stepping and position counter update will cease immediately without deceleration. Any program "running" will be terminated.

Any axis in the program mode will exit the program mode without creating the "end of program marker," therefore the escape character is useful in editing non-volatile program segments. In single mode a pound (#) sign is returned.

Note: Because the deceleration is immediate (without ramping) mechanical overshoot may result, especially with high speeds and/or inertia loads.

Command	Function		Type	NV Bytes
@	Soft Stop		Immediate, Program	1
	Mnemonic	**Data 1**	**Data 2**	**Result**
	(Name) @	None	None	#

@ (Soft Stop)

The Soft Stop command is useful as a gentle stop. It behaves differently, depending on how it is used. If the axis is moving it causes an immediate deceleration to a stop, based on the established deceleration K value. If one or more axis is running a program when this command is sent via the serial port, the running program(s) will terminate after deceleration. The soft stop may be embedded within a program (in Program mode). During program execution, the encountered soft stop will not cause termination and is functionally equivalent to the "M 0" command.

Command	Function		Type	NV Bytes
^C	Reset Controller		Immediate, Program	0
	Mnemonic	**Data 1**	**Data 2**	**Result**
	(Name) ^C	None	None	None

^C (Reset)

Software Reset is a global RESET command. All axis ABORT immediately and a reset, equivalent to the power-up condition, is executed:

1. Down load default values from the non-volatile memory.
2. Set origin(s) to zero.
3. Calibrate motor current to hold value.
4. Assume an idle state waiting for GO pulse input, Jog input or serial command input.

Command	Function		Type	NV Bytes
A	Read/Write to Ports		Immediate, Program	2
	Mnemonic	**Data 1**	**Data 2**	**Result**
	(Name)A (n)	0-129	None	Port Data

A (Port)

Input data ranging between 0 and 31 is complemented then output to port 1 through port 5. Port 1 is the least significant bit. Binary combinations of bits will turn on more then one port. Examp le; "A7" will set ports 1,2 and 3 to an ON condition. At hardware reset all outputs are set off (high). The command "A 128" will cause ports 1 through 5 to increment in a binary fashion. The command "A 129" will read and display the port data. The command "A0" will reset all ports to their in-active state.

Port	Data	Port	Data
1	1	Trip	64 Inverts Output
2	2	Increment	128 Increment port 1-5
3	4	Read	129
4	8		
5	16		

Reading the port data provides the following result information:

Data	Cause	Data	Cause
1	Low input present on port 1*	16	Low input present on port 5*
2	Low input present on port 2*	32	High if moving
4	Low input present on port 3*	64	Trip Point passed
8	Low input present on port 4*	128	Direction level, high for CW rotation

*Ports 1,2 and 3 are dedicated inputs. Port 4 is an input and output. Port 5 is a dedicated output. The ports are forced active low by the set port command. The "A0" command will reset all ports. Result: None/Read State

Command	***Function***		***Type***		***NV Bytes***
B	Set Jog Speeds		Immediate, Program		3
	Mnemonic	***Data 1***		***Data 2***	***Result***
	(Name) B (n1,n2)	Slow Speed 0-255		High Speed 0-255	None

B (Set Jog Speeds)

Data range 1 and data range 2 represent the speeds to use when the jog inputs are utilized. The first is usually a slower speed. The second number is used when the high speed jog (J2 pin 13) is held low. The values are multiplied by 30 to determine the actual step rate in steps per second. Setting values of "0" will disable the jog inputs. Ramped acceleration using the "K" value is implemented. Ramped deceleration is used if the high speed input is released while maintaining one of the two jog inputs on (low). Both clockwise and counter clockwise inputs operate at the same speeds.

The Jog inputs are active:

1. After power-up.
2. When not executing a motion command.
3. When not running a program.

Jogging is inhibited during actual command line entry.

<table>
<tr><td rowspan="4">Command

C</td><td colspan="2">Function</td><td colspan="2">Type</td><td>NV Bytes</td></tr>
<tr><td colspan="2">Clear and Restore</td><td colspan="2">Immediate</td><td>0</td></tr>
<tr><td>Mnemonic</td><td colspan="2">Data 1</td><td>Data 2</td><td>Result</td></tr>
<tr><td>(Name) C (n)</td><td colspan="2">0-8 Page</td><td>None</td><td>Version</td></tr>
</table>

C (Clear and Restore)

The C1 command clears the entire non-volatile memory and restores the default parameter settings. This command should be used sparingly since the non-volatile memory has a finite life of approximately 400,000 write cycles.

Parameter	Values (after Non-volatile Clear (C1) Command)
Initial velocity (I)	400 steps per second
Slew velocity (V)	5,016 steps per second
Divide factor (D)	1
Ramp slope (K)	5
Jog speeds (B)	30, 200 (x30 steps/sec)
Trip Point (T)	Off
Full/Half (H)	Full
Auto power down (E)	Yes
Power down polarity (E)	High
Hold/Run current (Y)	10, 75
Limit polarity (H)	Low
Auto position readout (Z)	Off
Name	Unchanged

All these parameters are saved as a block from the working registers. Frequent use of this command should be avoided, as memory longevity may be effected.

To complete the restoration of programmable current control for the motor load, the appropriate E command (typically E8) should be entered and saved (S) immediately following the use of the C1 command.

The following commands clear the corresponding page of non-volatile memory from the indicated address locations:

Command	Page#	Address
C2	2	256 to 511
C3	3	512 to 767
C4	4	768 to 1023
C5	5	1024 to 1279
C6	6	1280 to 1535
C7	7	1536 to 1291
C8	8	1792 to 2047

Command	Function		Type	NV Bytes
D	Divide Resolution		Immediate, Program	2
	Mnemonic	***Data 1***	***Data 2***	***Result***
	(Name) D (n)	Resolution 1-255	None	None

D (Divide Resolution)

All speeds during ramping and slewing are divided by the specified number (n). The prescale number may range between 1 and 255. Speeds as low as 3 steps per minute may be obtained. As "n" is increased, other parameters (internal speeds) must be increased to obtain a given output step speed. Using a value of 2 or 3 may be helpful in producing smoother acceleration characteristics at slower slew speeds. The "D" command should NOT be changed while moving.

Command	Function		Type	NV Bytes
E	Enable Current Control		Immediate, Program, Default	2
	Mnemonic	***Data 1***	***Data 2***	***Result***
	(Name) E (n)	0, 8, 12	None	

E (Enable Control)

The E command controls the limit switch activation polarity. It works in conjunction with the Y command to establish the driver Hold and Run current to the motor load.

E0: Disables the driver output.
E8: Enables the driver output based on the Y settings and is used for "normally open"* limit switch activation.
E12: Enables the driver output based on Y settings and is used for "normally closed" limit switch activation.

*Limit Switch Activation: A limit switch that is normally open is defined as a High to Low activation.

E8 is the most commonly used control mode for programmable current using normally open limit switches. Reference the "Y" command to individually program motor run and hold currents.

Command	Function		Type	NV Bytes
F	Find Home		Immediate, Program, Default	2
	Mnemonic	*Data 1*	*Data 2*	*Result*
	(Name) F (n,d)	SPS (18-23,000)	Direction (0-1)	None

F (Find Home)

The special Find Home algorithm is intended to eliminate mechanical hysteresis typically found in many switches and encoders. Home is always approached from the same direction based on the initial logic state of the Home switch and the value (0 or 1) assigned to the "d" direction byte.

1. The Find Home step velocity, using a normally open Home switch (actuation from logic high to low) is programmable over the entire slew velocity available; 18 to 23,000 SPS. Once the Home switch is encountered the system inertia typically overshoots the exact switch transition point so that the controller changes the direction signal and shifts the step speed down to the (I) initial parameter velocity. This direction reversal and speed reduction continues until the exact Home switch actuation point is reached and the Homing function is complete.

2. The Find Home step velocity, using a normally closed Home switch (actuation from logic low to high) will always be the (I) initial velocity parameter setting. Once the Home switch is actuated, all motion ceases and the Homing function is complete.

The following table illustrates the possible combinations of switch motion:

Home Switch	Direction Parameter	Direction of Motion
Normally Open (High to Low	0	Negative
Normally Closed (Low to High)	0	Positive
Normally Open (High to Low)	1	Positive
Normally Closed (Low to High)	1	Negative

Command	Function		Type	NV Bytes
G	Execute Program		Immediate, Program	3
	Mnemonic	*Data 1*	*Data 2*	*Result*
	(Name) G (a,t)	0-2048	None	None

G (Go)

The Go command executes a User programmed sequence starting at a predefined address location. Although most programs will start at "0," the User can start at another address. That address however, MUST begin at a stored instruction address. The Trace option is useful in debugging single axis programs. If trace is a 1, the Trace mode is turned on. A display of the current step being executed is produced while the program is running. The list format is the same as that of the "Q" command.

The Trace mode will be in effect until the program execution terminates or until an embedded Go without the trace attribute is encountered. Address locations between 225 and 255 are reserved for parameter storage and may not be used in programs.

There is a special case for the Go instruction. If the address is specified as 2048 (above the last non-volatile address), the control signal will branch to an address based on the state of input ports 1 through 4. The target address starts at the second page of program memory at address 256, with 16 character (byte) intervals. This instruction is analogous to "on PORT go to."

Input Port State				Address
P1	P2	P3	P4	Location
1	1	1	1	256
0	1	1	1	272
1	0	1	1	288
0	0	1	1	304
1	1	0	1	320
0	1	0	1	336
1	0	0	1	352
0	0	0	1	368
1	1	1	0	384
0	1	1	0	400
1	0	1	0	416
0	0	1	0	432
1	1	0	0	448
0	1	0	0	464
1	0	0	0	480
0	0	0	0	496

The physical input ports are internally inverted as part of the address computation. State 1111 is defined as a high or +5V on Port 1 through Port 4. Reference: BRANCH.SMC program structure on the EASI diskette.

Command	***Function***		***Type***	***NV Bytes***
H	Step Resolution, Decay, Calibrate Timing		Immediate, Program, Default	2
	Mnemonic	***Data 1***	***Data 2***	***Result***
	(Name) H (n)	Decay, Full/Half (0-3, 16)	None	None

H (Step Resolution, Decay, Calibrate Timing)

This command selects decay mode, step size resolution and timing calibration. Any User sequence in non-volatile memory is left intact; however a hardware reset or clear command (C1) MUST be used to access the User values. When changing between full and half step, position is maintained to the nearest full step. Default = 0 (full step).

Decay Mode

This feature provides for *FAST* and *SLOW* current decay modes, permitting you to select the best decay mode for your particular application.

Fast decay mode regulates motor current by varying the duty cycle and applying full negative DC bus voltage to the motor windings when OFF. Applying full DC bus voltage in this manner causes the motor winding current to change at a rapid rate; hence ‘Fast Decay.” The Fast Decay mode of operation permits better current regulation, but increases motor heating due to the higher current transients.

Slow Decay mode also regulates motor current by varying the duty cycle, but instead, shorts the motor windings when OFF (i.e. zero volts vs. maximum DC bus voltage).

The Slow Decay mode of operation permits higher currents (and thus higher torque) at lower speeds with less motor heating; but also exhibits poor low current regulation.

Settings

‘H’	Decay	Step Mode
H0	Slow	Full
H1	Slow	Half
H2	Fast	Full
H3	Fast	Half

Default = 0 (full step).

H 16

Earlier versions of the SMC-C24 (microprocessor) used in the SAX/DAX had an 11 Mhz clock and the value used in the “W” (Wait) command was correct. New devices are specified at 14.7 Mhz and the calibration is off by 4/5. Normally, the designer has compensated for this error by adjusting the data value. Software V2.07 provides for calibration to 10 Ms, while being backward compatible with earlier versions (8 Ms).

Setting calibration on:

1. Enter the command “H 16”. This sets only a calibrate flag and does not change other “H” modes (0-15). H (0 thru 15) does not affect the calibrate flag.
2. Issue the S (save) command. The mode is saved in NV memory.

The CLEAR (“C1”) command will cause the calibrate flag to be reset, and steps 1 and 2 must be repeated.

To determine the calibrate mode, the EXAMINE (X) command may be used. When in dumb terminal mode, a lower case “w” will appear as the first character of the display response.

Command	***Function***		***Type***		***NV Bytes***
I	Initial Velocity		Immediate, Program		3
	Mnemonic	***Data 1***		***Data 2***	***Result***
	(Name) I (n)	SPS 18-23,000		None	None

I (Initial Velocity)

The initial velocity specifies the start and stop speed for the motor. The first velocity of the acceleration ramp is specified by the I parameter. To achieve an optimum value the following variables must be taken into account: motor size (rotor inertia), system inertia, starting torque to overcome friction (stiction), and motor or system resonance’s. Generally values between 50 and 1200 SPS are appropriate.

The initial velocity applies to:

1. All index commands (+, -, R).
2. Start speed used in constant velocity (M).
3. Decelerate to 0 in constant velocity or soft stop.
4. Final phase of home routine.

The Examine (X) command displays velocity information. This parameter is a user default. After setting the value and issuing the Save command, it will be stored for future power-ups. A value of 400 SPS is preset on issuance of the “C8” (clear) command.

Command	Function		Type		NV Bytes
J	Jump to Address		Program		4
	Mnemonic	***Data 1***	***Data 2***		***Result***
	(Name) J (a,n)	Address (2047)	N + 1 Times (0-255)		None

J (Jump To Address)

Example: Jump to address a, n + 1 times.

This loop command allows repetition of a sequence up to 255 times. The address specified MUST be a valid instruction address and may be used only within a program. This instruction may NOT be nested. Only one jump counter is available for use at any given time.

Command	Function		Type		NV Bytes
K	Ramp Slope		Immediate, Program		2
	Mnemonic	***Data 1***	***Data 2***		***Result***
	(Name) K (n)	Accel (0-127)	Decel (128-255)		None

K (Ramp Slope)

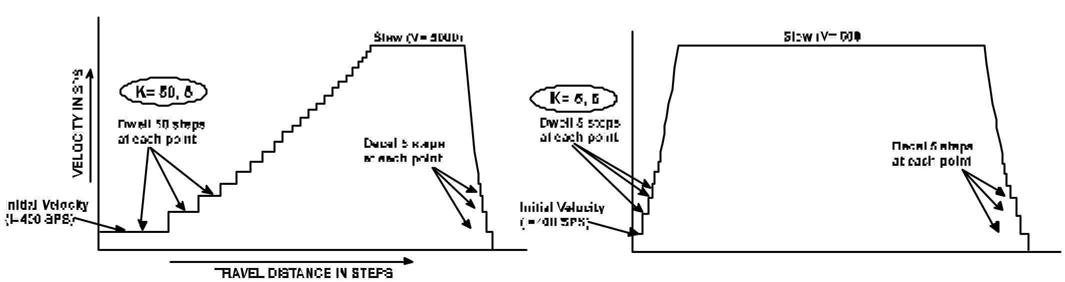
Specify the ramp acceleration and deceleration time.

The "K" command is used to adjust the ramp slope during the motor acceleration or deceleration. The profile or shape of the acceleration/deceleration curve is defined by an internal lookup table. Depending on the values of initial and slew velocities, 0 to 157 discrete velocities may be attained to adjust the acceleration or deceleration of the motor armature rotation.

The "K" value determines how many steps are made at each step rate point on the acceleration curve during ramping. Higher "K" values will increase the dwell time at each discrete point on the acceleration ramp. Lower values of "K" will increase the acceleration rate. A value of 0 will eliminate any ramping.

In practical applications, it is typically easier to decelerate a system, rather than accelerate a system. The separate decelerate parameter feature is a valuable time saver when compared to systems with fixed acceleration/deceleration times.

Examples: Two ramped indexes, each 2000 steps with I=400, V=5000, but different "K" values; K50 5 and K5 5.



Note: Values of "K" for deceleration mirrors the acceleration ramp. If a value of less then 127 is entered for "K", then both ramps assume the same slope. To alter the deceleration ramp, it is necessary to enter values between 128-255.

Example:
Entering a value of "130" would change only the deceleration slope and would cause the deceleration ramp to have a value of "2" steps on each point of the deceleration portion of the ramp table.

The K command can be issued:

1. As part of a setup.
2. In an application program.
3. As User defined defaults at reset.

Command	Function		Type		NV Bytes
L	Loop on Port		Program		3
	Mnemonic	***Data 1***	***Data 2***		***Result***
	(Name) L (a,c)	Address (0-2648)	Condition (0-9)		None

L (Loop On Port)

The Loop command will test the specified input port for the required condition . If the port is NOT at the required level, the program will jump to the specified address. If the address is to a previous instruction, the program will loop until it becomes the specified level. The program will then continue to the next step. Input ports are tested as follows:

Condition	Wait For
0	Port 1 Low
1	Port 1 High
2	Port 2 Low
3	Port 2 High
4	Port 3 Low
5	Port 3 High

There is also an additional feature for implementing a "wait until" function. The standard loop tests the condition every 2-3 Ms. If the unique address is 2048, the controller executes a tight loop at this instruction while monitoring the specified condition. When the condition is met, program execution continues. This feature is helpful in situations where the condition may be of short duration. This command is usable only in non-volatile program execution.

Command	Function		Type		NV Bytes
M	Move at Constant Velocity		Immediate, Program		3
	Mnemonic	***Data 1***	***Data 2***		***Result***
	(Name) M (±)	SPS ± 23,000	None		None

M (Move At Constant Velocity)

Move at the specified velocity and direction “forever."

The motor will ramp up or down to a constant step velocity and motion will continue at the given speed until a new velocity is entered. The specified slew speed is in steps per second. Ramp parameters (K command) may be modified prior to each velocity command, allowing different ramp slopes.

The direction is specified by the sign preceding the velocity. Decelerating from full speed in one direction to full acceleration speed in the opposite direction can be accomplished with this single command.

Examples:
M 1000

Move in the + direction at 1000 SPS starting at the initial velocity then accelerating to the 1000 SPS slew speed.
M -4000

Decelerate to initial velocity (from 1000). Stop and change direction. Accelerate to 4000 SPS in the “-” direction.

Motion may be stopped by:
1. The "M 0" command.
2. Soft stop "@" command or interrupt.
3. ABORT (ESC) interrupt (without deceleration).

The default initial velocity is used at the first invocation of the command. Position Trip points may be used. If the encoder feedback option is implemented, stall supervision is functional.

Command	*Function*		*Type*	*NV Bytes*
O	Set Origin		Immediate, Program	4
	Mnemonic	*Data 1*	*Data 2*	*Result*
	(Name) O (±n)	Position (± 8,388,607)	None	None

O (Set Origin)

The Set Origin command resets the internal 24 bit position counter to the specified value. Base position for the Relative mode is zero. Signed numbers are used. Hardware reset clears to zero.

Command	*Function*		*Type*	*NV Bytes*
P	Program Mode On/Off		Immediate	0
	Mnemonic	*Data 1*	*Data 2*	*Result*
	(Name) P (a)	Address (256-2047)	None	None, #

P (Program Mode)

This Program command allows composition and storage of “program sequences" in non-volatile memory. Program mode allows entering commands for future execution by use of the Go command or External go signal. Existing programs are overwritten as new instructions are stored. Entering a second "P" command will terminate the Program mode, and insert an end of program marker in the stored program before returning to the Immediate Command mode. While in Program mode, commands and data are directed into the non-volatile memory. The address specifies the start point in non-volatile memory where the application program will reside. As instructions are entered, the address counter is updated and displayed. Any number of independent program segments can coexist. These can be accessed via Jump, Loop, Go or other special instructions.

Special Locations, "Shadow Memory"

Sixty four bytes of the program are configured as FAST memory. Locations 128 thorough 192 are downloaded from the external EEprom to internal RAM at power-up.

Instructions executing in this segment run faster then other locations that fetch from relatively slow EEprom (1 Ms. per byte). Programmers should reserve this segment for time critical code. The shadow RAM is not actually written to non-volatile memory until the Store command is issued. Host computers may download subroutines without concern about wearing out the EEprom. Locations 256 through 511 are predefined if the "G 2048" command is used in an application.

The Trip command can jump to any location between 0 and 255. Editing of programs should be done WITH CAUTION. The general programming sequence is to (1) start programming at desired address, (2) enter new instructions, and (3) terminate programming with the ESC command. (This will cause a return to the Command mode without inserting the end of program marker).

Note: Do not attempt to use locations above 1792 for programs (default storage).

Example program in terminal mode (each command is terminated with Carriage return):

Response	Command	Comment
	P 0	Enter program mode.
0 (echoed)	+5000	Index 5000 steps.
5	-5000	Index -5000 steps.
10	P 0	End program, insert program marker One program has been entered and stored in non-volatile memory. You may now insert another routine.
	P 100	Start program at memory location 100.
100 (echoed)	O	Set origin to zero.
101	R 2000	Move to position 2000.
106	R - 500	Move to position –500.
111	J 101 9	Repeat sequence 10 times.
	P 0	End program.You may now list the programs.
	L 0 5	
	L 100 5	You may now execute the programs.
	G 100	You may now edit the programs.
101	R 3000	Change position from 2000 to 3000.
	P 101	
106	<ESC>	Terminate edit without "end" mark.

While most commands can be part of a program, some may not be necessary or impractical while some should just be avoided.

Command	Reason
B Jog Speeds	User default from non-volatile memory at power-up
C Clear	Never use - it will erase the programs
D Resolution	User default fromnon-volatile memory at power-up
E Delay	User default from non-volatile memory at power-up
H Resolution Mode	User default from non-volatile memory at power-up
I Initial Velocity	User default from non-volatile memory at power-up
K Ramp Slope	User default from non-volatile memory at power-up
Q Query	N/A
S Store	No application, non-volatile memory wear
T Trip	User default from non-volatile memory at power-up
V Slew Speed	User default from non-volatile memory at power-up
X Examine	N/A
Y Current Control	User default from non-volatile memory at power-up

Command	Function		Type	NV Bytes
Q	Query (List) Program		Immediate	0
	Mnemonic	*Data 1*	*Data 2*	*Result*
	(Name) Q (a)	Address (0-2047)	None	Listing

Q (Query Program)

This command will produce a list (disassembled) of the instructions stored in non-volatile memory using the format: "ADDRESS" "INSTRUCTION" "DATA 1" "DATA 2"

1. The values will be displayed only if applicable to the particular instruction type.
2. One instruction will be listed at a time.
3. The space key will advance the address and list more instructions.
4. Listing is terminated when an "end of program" marker is found or the ESC character is received.

Command	Function		Type	NV Bytes
R	Index Relative to Origin		Immediate, Program	4
	Mnemonic	*Data 1*	*Data 2*	*Result*
	(Name) R (dd)	Position (± 8,388,607)	None	None

R (Index Relative To Origin)

Move, with ramping, relative to the “0” origin. The target position has a range of ±8,388,607 steps from the ‘0’ origin.

The motion sequence is:
1. Wait until any previous motion is finished,
2. Read the current position then calculate the distance to the new target position,
3. Energize the motor winding,
4. Start stepping at the rate of the initial velocity (I),
5. Accelerate using a profile defined by the fixed table that approximates straight-line acceleration and a slope set by the “K” command,
6. The acceleration continues until the slew speed as specified by the “V” command is attained,
7. Motion continues at the slew speed, until the deceleration point is reached,
8. Decelerate (determined by the second “K” value) to a stop completing the index,
9. If another index is not commanded for the settling period, power down the motor (if auto power down is enabled).

Command	Function		Type	NV Bytes
S	Store Parameters		Immediate, Program	1
	Mnemonic	*Data 1*	*Data 2*	*Result*
	(Name) S	None	None	None

S (Store Parameters)

The S (Store) command will force a write sequence to non-volatile memory. This command should be used to save new operational parameters or to down-load motion programs. The S command should not be embedded internally as part of a program but rather used to save the program after loading.

The following parameters are saved in the non-volatile memory and will be recalled as defaults during power-on reset:

Parameter	Standard Defaults
Initial velocity (I)	400 SPS
Slew velocity (V)	5,016 SPS
Divide factor (D)	1
Ramp slope (K)	5
Jog speeds (B)	30, 200
Trip Point (T)	Off
Full/Half (H)	Full
Auto power down (E)	Yes
Power down polarity (E)	High
Quarter step (H)	Off
Hold/Run current (Y)	10, 75
Limit polarity (H)	Low
Auto position readout (Z)	Off
Name (after reset)	Undefined

All of these parameters are saved as a block from the working registers in the SMC-C24. Frequent use of this command should be avoided, as memory longevity may be affected.

Note: Observe the precaution of writing too frequently to non-volatile memory.

Command	***Function***		***Type***	***NV Bytes***
T	Set and Enable Trip Point		Default, Program	4
	Mnemonic	***Data 1***	***Data 2***	***Result***
	(Name) T (n)	Position (± 8,388,607)	None	None

T (Trip Point)

The SAX/DAX have a programmable "Trip Point" feature. During moves the current position is compared to the Trip position at each step. The Trip output will be alternated each time the Trip Point is reached or passed. When in the Program Run mode, a User program located at address 200 will be automatically executed (called). On completion of this sub program, a return to the call point will be executed.

When the Trip Point is enabled, available top step speed capability is reduced by approximately 10%. A value of 0 will disable the Trip Point trigger. A value of -0 (minus zero) will set the Trip Point to 0. The Trip Point is displayed using the examine command. Default = 0 (off).

Command	***Function***		***Type***	***NV Bytes***
V	Set Final (Slew) Velocity in SPS		Default, Immediate, Program	3
	Mnemonic	***Data 1***	***Data 2***	***Result***
	(Name) V (n)	SPS (18-23,000)	None	None

V (Slew Velocity)

This command sets final Slew Velocity after ramping up to speed. The final output velocity is divided by the value of "D." This value is independent of constant velocity (ramp completed), jog or home speeds and is used for indexing absolute or relative.

The following commands use this parameter:
R (Relative index)
+ (Plus index)
- (Minus index)

The following functions do not use or affect this parameter:
J (Jog)
F (Find home)

The SAX/DAX also allows specifying speeds in step time values. The actual step clock period may be directly controlled, rather than in steps per second. This allows fine control of step rates, especially at lower speeds. The SAX/DAX is placed in this mode by using a negative value with the “V” command, and restored to the SPS mode with positive values. Switching between modes will not effect speeds previously specified. In the period mode, all subsequent commands that relate to speed must use period values for data (positive data).

These include:
V (-65535 to -54) Slew velocity
I (65535 to 54) Initial velocity
M (65535 to 54) Constant velocity
F (65535 to 54) Find home
B Jog Speeds (minimum obtainable value is 160 SPS)

The data range is between 54 (23,000 SPS) and 65,535 (18.76 SPS).

The formula for determining step rate is:
N = 1228800 / SPS

As with all speed commands, the output rate is divided by the value specified with the “D” command.

Command	Function		Type		NV Bytes
W	Wait (n) Milliseconds		Immediate, Program		3
	Mnemonic	**Data 1**		**Data 2**	**Result**
	(Name) W (n)	0.0075 Sec. (0-65.535)		None	None

W (Wait)

WAIT n milliseconds. (See related “H16” command)

The controller will remain in an idle state for the specified amount of time. The Wait command, if issued while indexing (as a result of a R,+,- or F command), will NOT start until all motion is complete. Using this command with 0 time can provide an alternate method of determining motion. If issued while running at constant velocity, the time out will occur without waiting for motion to cease.

High speed step operations, executed during wait commands, will increase the delay time by as much as 14 times normal value. The result will NOT be available until the delay is complete.

Command	Function		Type	NV Bytes
X	Examine Settings		Immediate	0
	Mnemonic	***Data 1***	***Data 2***	***Result***
	(Name) X	None	None	Display Setting

X (Examine Parameters)

The Examine command displays the parameter settings. This command will produce two different responses, depending on the mode of Motion command operation. When in the non-Party Line (Single) mode the display is as follows:

(C=cc) K=kk, I=ii, V=vv, (T=tt) nn [Carriage Return, LF]"

Where:
cc = Phase current setting (if enabled and will display Hold or Run value depending on operational mode)
kk = Ramp factor
ii = Initial velocity divided by "D" (calculated)
vv = Slew velocity divided by "D" (calculated)
tt = Trip Point (if enabled)
nn = Name of Party Line

When in the multi-axis (Party Line) mode the data is returned in the following format:

cc kk ii vv tt (LF). It is assumed the host computer will interpret the data string for processing.

Command	Function		Type	NV Bytes
Y	Set Hold and Run Current		Immediate, Program	3
	Mnemonic	***Data 1***	***Data 2***	***Result***
	(Name) Y (h,r)	Hold (0-100)	Run (0-100)	None

Y (Hold and Run Current)

This command allows specifying the Hold and Run values of motor current (per phase) between 0 and 100% with a resolution of 1%. To utilize this feature the following commands must have been initiated:

1. The Current Control mode is enabled via a E8 or E12 Enable command.
2. The Y (xx) (xx) data values were entered.
3. A store command was issued (S).

Step 1 and step 2 need only be executed once to establish the Current Control mode, unless the C1 Clear and Restore command was used.

The two values may be saved in non-volatile memory by use of the "S" (Save) command. Current change does not take place until a subsequent "E" command is issued or a motion command is executed.

Refer to the Addendum: “About Step Motor Current” for more information.

Command	Function		Type	NV Bytes
Z	Read and Display Current Position		Immediate, Program	2
	Mnemonic	***Data 1***	***Data 2***	***Result***
	(Name) Z	Readout Mode (0-1)	None	None

Z (Read Position)

Read and display the current position. During motor move commands the value will change depending on the direction of travel. The position counter is reset by the "O" (Set Origin) command. The "Z1" command enables a continuous readout, via the serial interface. Any change in position causes the position data to be sent to the serial output. The readout is terminated by a Carriage Return only. The Readout mode will be defaulted "on" if a Save command is issued. This mode is only practical using a single axis protocol.

Command	Function		Type	NV Bytes
[	Read NV Memory		Immediate	0
	Mnemonic	***Data 1***	***Data 2***	***Result***
	(Name) [	Address (0-2047)	Number (0-255)	Displayed Values

[(Read Non-Volatile Memory)

The Read Non-Volatile Memory command allows the User to display any byte of the 2047 byte external non-volatile memory. The address specifies the desired location to access.

Command	Function		Type	NV Bytes
]	Read Limits, Hardware		Immediate, Program	1
	Mnemonic	***Data 1***	***Data 2***	***Result***
	(Name)]	0-1	None	None

] (Read Limits/Hardware)

This command allows examination of the status of the various switch inputs. The result will contain the state of the limit switch inputs and current phase outputs in binary values, as follows:

Decimal Value	128	64	32	16	8	4	2	1
Bit Position	7	6	5	4	3	2	1	0
Controller	Lb	La	Hm	*	Qs	Fb	En	Half

Where:

- La= Limit "a" switch
- Lb= Limit "b" switch
- Hm= Home switch (32=low input)
- *= Always low
- Qs= Quarter step output level (n/a: SAX/DAX)
- fb+,fb-= Encoder feedback input level (n/a: SAX/DAX)
- En= Enable output level
- Half= Half step output level

If the motor is disabled and limit switches are inactive the result will be 0.

Command	Function		Type		NV Bytes
+	Index in the Plus Direction		Immediate, Program		4
	Mnemonic	***Data 1***	***Data 2***		***Result***
	(Name) + (n)	Steps (0-16,777,215)	None		None

+ (Index In Plus Direction)

Step in a positive direction for the specified distance. The motor will ramp up, slew, then ramp down per the previously set parameters. The range is 0 to 16,777,215 steps. The position counter will overflow at 8,388,607. A "+" direction index is defined as a clock-wise rotation of the motor as viewed looking toward the motor mounting flange.

Command	Function		Type		NV Bytes
-	Index in the Minus Direction		Immediate, Program		4
	Mnemonic	***Data 1***	***Data 2***		***Result***
	(Name) - (n)	Steps (0-16,777,215)	None		None

- (Index in Minus Direction)

Step in a negative direction. Except for the direction, this command behaves exactly as the "+" command above.

A "-" direction index is defined as a counter clock-wise rotation of the motor as viewed looking toward the motor mounting flange.

Command	Function		Type		NV Bytes
^	Read Moving Status		Immediate, Program		1
	Mnemonic	***Data 1***	***Data 2***		***Result***
	(Name) ^	None	None		Status

^ (Read Moving Status)

The Read Moving Status command is used to determine the current moving and mode status. These status bits are converted to a decimal number (0-255) in Terminal mode.

The status byte contains the current status of the controller as follows:

Bit	Decimal	Status
0	1	Moving - indicates axis moving
1	2	Constant - high in constant velocity
3	8	Homing - homing routine is active
4	16	Slewing - ramping complete

Note: Other bits in this result should be ignored.

<table>
<tr><td rowspan="4">Command

\</td><td colspan="2">Function</td><td>Type</td><td>NV Bytes</td></tr>
<tr><td colspan="2">Write to Non-Volatile Memory</td><td>Immediate</td><td>0</td></tr>
<tr><td>Mnemonic</td><td>Data 1</td><td>Data 2</td><td>Result</td></tr>
<tr><td>(Name) \ (a,d)</td><td>Address (0-2047)</td><td>Data (0-255)</td><td>None</td></tr>
</table>

\ (Write To Non-Volatile Memory)

The Write to Non-Volatile Memory command allows the programmer to directly modify any byte in the memory. The life expectancy of the non-volatile memory may be effected by excessive use of this command.

Note: Non-volatile memory has a finite life of approximately 10 years for data retention and 460,000 write cycles.

Electrical

Power Supply...............................40 Volts @:
SAX-4X4................................. 45VA
SAX and DAX........................... 90VA
AC Fuse:
SAX-4X4…………………….. 1.0A @ 120V, .75A @ 240V; 2AG/SB
SAX and DAX…………….. 1.5A @ 120V, 1.0A @ 240V; 3AG/SB
Output Current (Peak)..................2.8 Amps
Chopping Frequency....................20kHz
Input Voltage...............................100 to 125 VAC, 60 Hz or 200 to 250 VAC, 50Hz
Motor Step Resolution.................Full/Half Step
Non-Volatile Memory..................2k Bytes
Position Counter..........................±8,388,607
Baud Rate.....................................9600 (Standard.), 38.4k (Selectable)

Output Signals

Parameter	Condition	Min	Typ	Max	Units
High Level (Voh)			5	15*	Vdc
Low Level (Vol)	Iol = 1.6			.45	Vdc
Low Current (Iol)				30	ma

* Open collector with internal pull-up to Vcc, over driven.

Input Signals

Parameter	Min	Typ	Max	Units
High Level (Vih)	2.4		5	Vdc
Low Level (Vil)	GND-0.5		0.8	Vdc
Low Current (Vlh)			2	ma

Optical Inputs

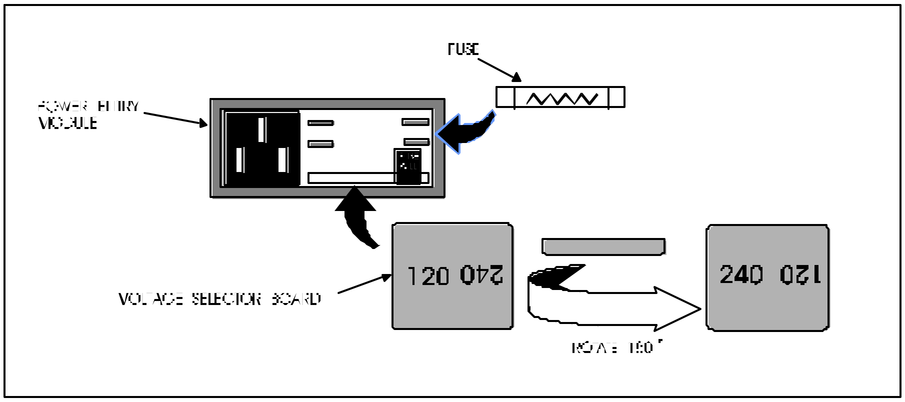
Parameter	Min	Typ	Max	Units
LED Current			5	ma
LED Drop			1.5	Vdc
Internal Series Resistor			1k	Ohms
RS-422 Input Level		5	25	Vdc
RS-422 Outputs		5	25	Vdc
RS-422 Loads*	2		32	Axes

The RS-422 implements drivers rated for EIA RS-485 transmission and limits receiver count to 32 with a maximum cable length of 4,000 feet. RS-232 units are rated for one receiver and a maximum cable length of 50 feet.

Power Entry Module

Primary AC Voltage Selection/Changing Fuse (not available on SAX-4X4)

The A.C. input voltage is factory set at either 120 Volts-50/60 Hz or 240 Volts-50/60 Hz, based on application requirements at time of order. To change either the fuse or the input operating voltage use the following procedure:



1. Remove the power cord from the power entry module.
2. Slide the fuse/voltage selector cover window all the way to the left.
3. Pull the "Fuse Pull" tab out and to the left and remove fuse.
4. If 240 Volt power is to be used, pull out the voltage selector board, located below the fuse mounting clips.
5. Select 240 Volt operation by orienting the PC board to read "240" on the top left side of the board.
6. Seat the voltage selector board back in position, and return the fuse to the mounting clips. Slide the cover window all the way to the right.

Note: 120 Volt and 240 Volt selection are the only valid positions on the selector PCB. 100 Volt and 220 Volt positions are NOT supported selector positions.

Environmental

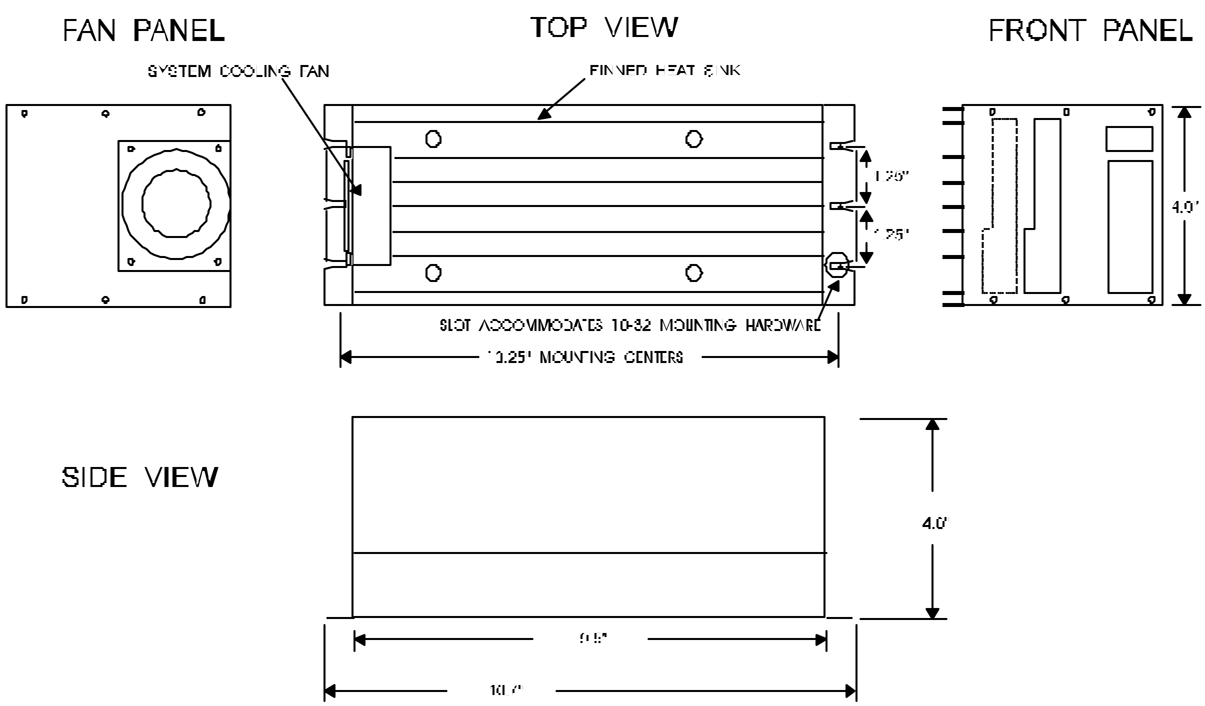
Storage...-45 to 85 Degrees C
Operating......................................0 to 55 Degrees C
Humidity......................................0 to 95% (Non-condensing)

Physical

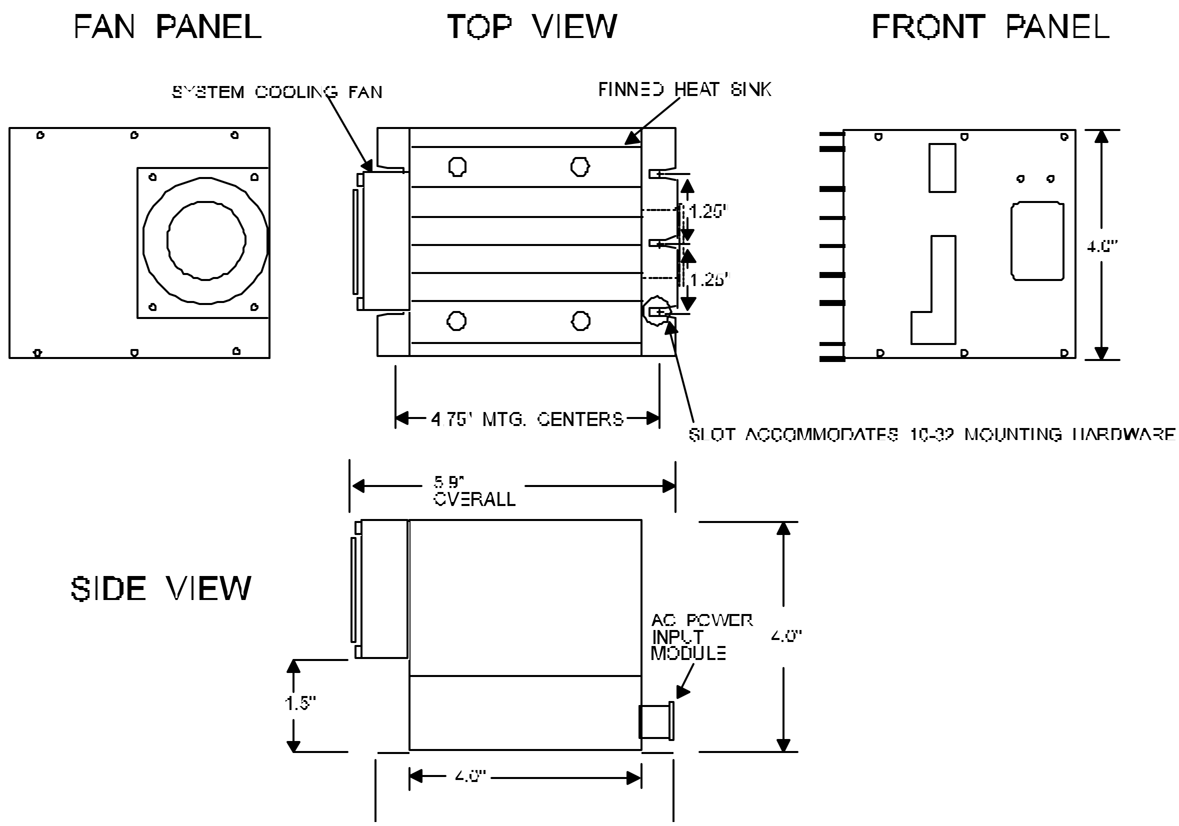
Size:
SAX-4X4..................................……… 4" W x 4" L x 4" H
SAX and DAX..................……… 4" W x 10.7" L x 4" H

Weight:
SAX-4X4...........................………… 3 lbs. 2 oz. (1.4 kg)
SAX.................................………… 6 lbs. 0 oz. (2.73 kg)
DAX..................................……... 6 lbs. 8 oz. (3.0 kg)

SAX/DAX Dimensional Drawing



SAX-4X4 Dimensional Drawing



Command Summary

MNEMONIC / COMMAND		DATA 1	RANGE	DATA 2	RANGE	NV	D	M	P	U
+	INDEX IN "+" DIRECTION	STEPS	0-16777215			4		⊙	⊙	
-	INDEX IN "-" DIRECTION	STEPS	0-16777215			4		⊙	⊙	
ESC	ABORT/TERMINATE									⊙
@	SOFT STOP					1			⊙	⊙
^C	SOFTWARE RESET									⊙
[	READ NV MEMORY	ADDRESS	0-2047	NUMBER	0-255					⊙
\	WRITE TO NV MEMORY	ADDRESS	0-2047	DATA	0-255					⊙
]	READ LIMITS/HARDWARE	LIM/HW	0-1			1				⊙
^	READ MOVING STATUS					1				⊙
A	PORT	BINARY	0-129			2			⊙	⊙
B	SET JOG SPEEDS	SLOW	0-255	HIGH	0-255	3	⊙		⊙	
C	CLEAR AND RESTORE	PAGE	1-8				⊙			
D	DIVIDE RESOLUTION	RES.	1-255			2	⊙		⊙	
E	ENABLE CONTROL	N VALUE	0,8,12			2	⊙		⊙	
F	FIND HOME	SPS	18-23000	DIRECTION	0-1	3		⊙	⊙	
G	GO	ADDRESS	0-226,256-2048	TRACE	0-1	3			⊙	
H	DECAY, RES./CALIBRATE	FULL/HALF	0-3			2	⊙			
I	INITIAL VELOCITY	SPS	18,23000			3	⊙		⊙	
J	JUMP	ADDRESS	0-225,2047	N+1 TIMES	0-255	4			⊙	
K	RAMP SLOPE	ACCEL	0-127	DECEL	128-255	2	⊙		⊙	
L	LOOP ON PORT	ADDRESS	0-2048	CONDITION	0-9	3			⊙	
M	MOVE AT CONST. VEL.	SPS	±18-23000			3		⊙	⊙	
O	SET ORIGIN		±8388607			4			⊙	⊙
P	PROGRAM MODE	ADDRESS	0-226,256-2048							
Q	QUERY PROGRAM	ADDRESS	0-2047							
R	INDEX TO POSITION	POSITION	±8388607			4		⊙	⊙	
S	STORE PARAMETERS					1	⊙			
T	TRIP POINT	POSITION	±8388607			4	⊙		⊙	
V	SLEW VELOCITY	SPS	18-23000			3	⊙		⊙	
W	WAIT, (DELAY)	0.0075 SEC	0-65535			3			⊙	⊙
X	EXAMINE PARAMETERS									⊙
Y	RUN AND HOLD CURRENT	HOLD	0-100	RUN	0-100	3	⊙		⊙	
Z	READ POSITION	CONTINUOUS	0-1			2				⊙

ASCII Character Code

Ctrl	Char	Dec	Hex	Code		Dec	Hex	Char		Dec	Hex	Char		Dec	Hex	Char
^@		00	00	NUL		32	20			64	40	**@**		96	60	**`**
^A	**?**	01	01	SOH		33	21	**!**		65	41	**A**		97	61	**a**
^B	**?**	02	02	STX		34	22	**“**		66	42	**B**		98	62	**b**
^C	**?**	03	03	ETX		35	23	**#**		67	43	**C**		99	63	**c**
^D	**?**	04	04	EOT		36	24	**$**		68	44	**D**		100	64	**d**
^E	**?**	05	05	ENQ		37	25	**%**		69	45	**E**		101	65	**e**
^F	**?**	06	06	ACK		38	26	**&**		70	46	**F**		102	66	**f**
^G	•	07	07	BEL		39	27	**‘**		71	47	**G**		103	67	**g**
^H	**?**	08	08	BS		40	28	**(**		72	48	**H**		104	68	**h**
^I	**?**	09	09	HT		41	29	**)**		73	49	**I**		105	69	**i**
^J	**?**	10	0A	LF		42	2A	*****		74	4A	**J**		106	6A	**j**
^K	**?**	11	0B	VT		43	2B	+		75	4B	**K**		107	6B	**k**
^L	**?**	12	0C	FF		44	2C	**,**		76	4C	**L**		108	6C	**l**
^M	**?**	13	0D	CR		45	2D	-		77	4D	**M**		109	6D	**m**
^N	**?**	14	0E	SO		46	2E	**.**		78	4E	**N**		110	6E	**n**
^O	¤	15	0F	SI		47	2F	**/**		79	4F	**O**		111	6F	**o**
^P	**?**	16	10	DLE		48	30	**0**		80	50	**P**		112	70	**p**
^Q	**?**	17	11	DC1		49	31	**1**		81	51	**Q**		113	71	**q**
^R	**?**	18	12	DC2		50	32	**2**		82	52	**R**		114	72	**r**
^S	**?**	19	13	DC3		51	33	**3**		83	53	**S**		115	73	**s**
^T	¶	20	14	EC4		52	34	**4**		84	54	**T**		116	74	**t**
^U	§	21	15	NAK		53	35	**5**		85	55	**U**		117	75	**u**
^V	**?**	22	16	SYN		54	36	**6**		86	56	**V**		118	76	**v**
^W	**?**	23	17	ETB		55	37	**7**		87	57	**W**		119	77	**w**
^X	**?**	24	18	CAN		56	38	**8**		88	58	**X**		120	78	**x**
^Y	**?**	25	19	EM		57	39	**9**		89	59	**Y**		121	79	**y**
^Z	**?**	26	1A	SUB		58	3A	**:**		90	5A	**Z**		122	7A	**z**
^[	**?**	27	1B	ESC		59	3B	**;**		91	5B	**[**		123	7B	**{**
^\	**?**	28	1C	FS		60	3C	<		92	5C	\		124	7C	|
^]	**?**	29	1D	GS		61	3D	=		93	5D	**]**		125	7D	**}**
^^	**?**	30	1E	RS		62	3E	>		94	5E	^		126	7E	~
^_	**?**	31	1F	US		63	3F	**?**		95	5F	_		127	7F	

About Step Motor Current

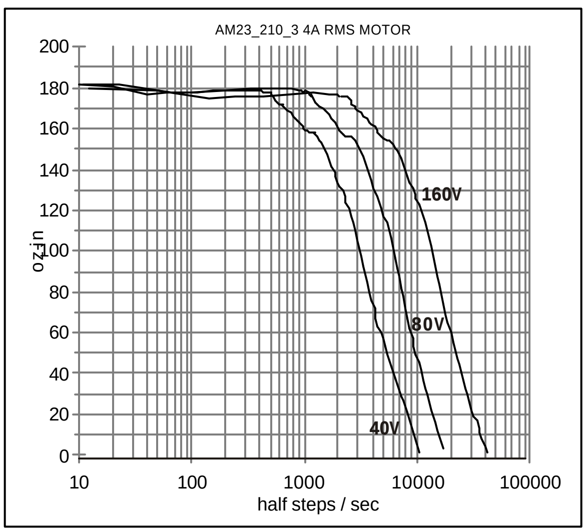
There is much confusion regarding the operation of step motors. Depending on your application, the step motor offers several advantages over servo motor designs, including lower cost and simplicity. The step (or stepper, or stepping) motor is a digital "synchronous" motor with a pre-designed number of "steps" per revolution. The most common motor has 200 full steps per revolution. Simple driver electronics can subdivide these steps into ½ step or more complex "microsteps."

Step Motor Characteristics

- The positional repeatability of each full or half step is very close to exact.
- While microsteps are repeatable, they tend to be somewhat non-linear.
- The torque is maximum at zero speed.
- The motor shaft RPM exactly correlates with the steps-per-second.
- Torque decreases with speed, eventually to zero or a "stall" condition.
- Resonance at certain speeds can cause undesired stalls or erratic operation.

There is little difference between today's step motor and the first generation of 60+ years ago. The magnetic materials and torque have been improved, yet it remains a simple, reliable workhorse of industrial motion control. Over time most improvements have been made to the drive and control electronics, i.e., microstep, solid-state components with higher voltage, current and switching speeds.

One insatiable hunger of a step motor is torque output at higher speeds. Winding inductance is the villain that limits speed. As the windings are switched on, the magnetic flux must be built up from current flow in the windings, producing mechanical torque. Higher step rates reduce the time available for flux to buildup and average current flow is reduced.



This reduced current results in reduced torque. The rate of current change depends on the voltage applied across it. High voltage applied across the coil will shorten the time constant.

Today's systems strive for low inductance motors and high voltage supplies. The above curves show the increased speed that might be obtained with higher supply voltages, up to 160Vdc. At standstill the average motor voltage is regulated to approximately 3Vdc.

A current sense circuit is used to switch off the current when it reaches the set value; hence the motor power is regulated. These "chopper" circuits operate at speeds above 20khz, well above hearing limits.

The following is an abstract from "Control of Stepping Motors, a Tutorial" (linked from www.stepcontrol.com) by Douglas W. Jones, University of Iowa Department of Computer Science. http://www.cs.uiowa.edu/~jones/step/index.html.

"Small stepping motors, such as those used for head positioning on floppy disk drives, are usually driven at a low DC voltage, and the current through the motor windings is usually limited by the internal resistance of the winding. High torque motors, on the other hand, are frequently built with very low resistance windings; when driven by any reasonable supply voltage, these motors typically require external current limiting circuitry."

"There is good reason to run a stepping motor at a supply voltage above that needed to push the maximum rated current through the motor windings. Running a motor at **higher voltages** leads to a faster rise in the current through the windings when they are turned on, and this, in turn, leads to a **higher cutoff speed** for the motor and **higher torques** at speeds above the cutoff."

"Microstepping, where the control system positions the motor rotor between half steps, also requires external current limiting circuitry. For example, to position the rotor 1/4 of the way from one step to another, it might be necessary to run one motor winding at full current while the other is run at approximately 1/3 of that current."

Motor Choice

The discussion here relates to bipolar chopper motors. Internally, standard motors have 4 windings, resulting in a total of 8 wire leads. Motor manufacturers supply various configurations:

Leads	Application Connection	Comment
8	Bipolar (series or parallel), unipolar	All 8 leads are available. External interconnect can be cumbersome and untidy.
6	Unipolar or bipolar series	Can be used with 50% copper reduced torque but increased speed possible.
4	Bipolar series or bipolar parallel	Series: higher torque but reduced speed capability. Parallel: higher speed with lowered torque.
5	Unipolar only	Not suitable for bipolar drives. See AMS model CCB-25 with programmable phase sequencing.

Determining the Current Value

Question: What is the right current value?
Answer: The minimum value to operate reliably.

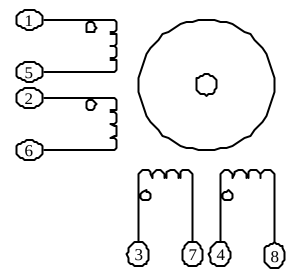
As the step motor current is reduced below the rated current, the torque output is reduced and eventually the motor will stall. The ideal current setting minimizes heating of motor and electronics, increases reliability, and reduces power supply requirements. Motors run more quietly and resonance effects can be reduced. One drawback from low current operation is that some microstep size linearity may be reduced, but full or half step accuracy is not adversely affected.

AMPS and Wire Count and Power

The rated current is specified based on the rated power input (watts) of a given motor.

A. Basic 8 Wire Motor

While never actually used as 8 individual coils, virtually all permanent magnet motors have 4 internal coils. All common configurations can be constructed from the 8 wire mo tor.



A Basic 8 Wire Motor

Let us assume that each winding of the 8 wire motor has the following specifications:

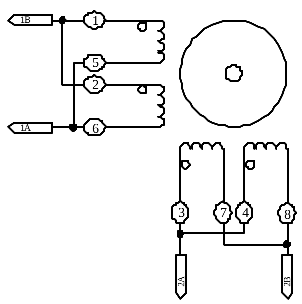
Current = 2 amps
Resistance =1.0 ohm
Voltage= 2.0 volts
Inductance = 4.4 mH

The power per winding is:

I^2R or 2 x 2 x 1= 4 watts,
x 4 coils = 16 watts total for this motor.

These values correspond closely with a NEMA size 23, 4 wire motor designs.

These following examples will configure the basic 8 wire motor into four real life connections:



4 Wire Parallel

4 Wire Parallel

The high-speed model implements parallel coil connection. Two coils connected in parallel result in the following for each of the two phases:

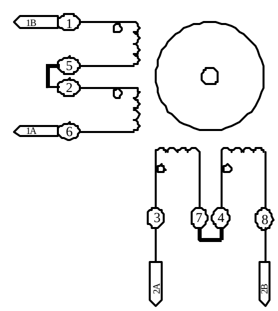
Parallel Resistance= 0.5 ohms
Parallel Inductance= 2.2 mH
Current= 4 amps (2 volts)
Watts per phase=8 (x 2 phases) = 16 watts total

B. 4 Wire Series

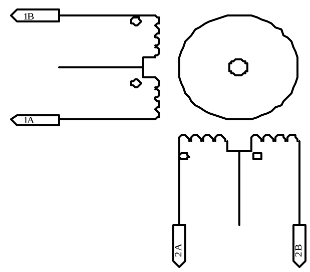
Changing to a series design, we have two pairs of two coils connected in series. Each has:

Series Resistance= 2 ohms
Inductance= 17.5 mH
The rated current is now 2 amps (4 Volts)
Watts per phase=8 (x 2 phases) = 16 watts total

Note that the series inductance is FOUR times the parallel design. Inductance limits the obtainable speed, since the time constant limits the amount of flux (hence torque) when step-to-step time is short.



4 Wire Series



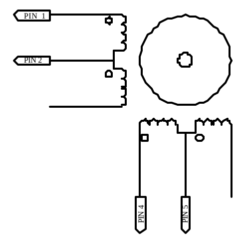
6 Wire Motor

C: Adapting Available 6 Wire Motors
A 6 wire motor is equivalent to the 4 wire series motor.

Series Resistance= 2 ohms
Inductance= 17.5 mH
Rated current= 2 amps (4 volts)

In practice the two coil ends are connected, while no connection is made to the center tap.

Half Copper or 50% Winding
The maximum speed can be increased by using ½ the coil. To do this, connect the driver between the center tap and one end of the winding.



6 Wire, Half Copper

The tradeoff is a loss of torque. The RMS current is the manufacturer's unipolar amperage rating with the same wattage per phase.

Often a 6 wire design is being upgraded or the size, features, availability or cost dictate the 6 wire motor. Some characteristics can make the motor impossible to use. Many motors are rated at voltages in excess of 5 volts. This means that 10 volts is necessary in the series (100% copper) configuration.

Aside from having excessive inductance, proper chopper operation dictates operation from voltage sources much higher than the motor rating. The minimum recommended value for VMM (DC supply) is 2 times the winding rating (the higher the better, until excessive heating occurs or insulation breakdown).

The RMS current rating for series operation is:

The manufacturer's unipolar amperage rating divided by 1.414. The lower current will reduce the average voltage slightly (about 7 volts).

Summary

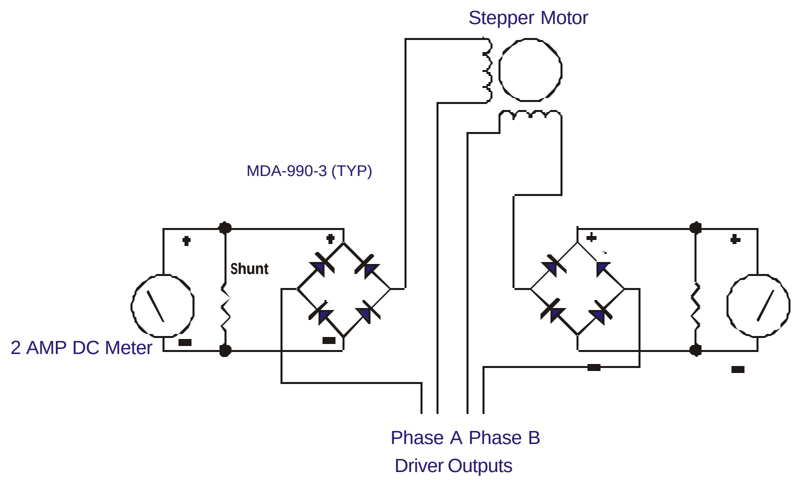
# Leads	Winding Connection	Per Phase Bipolar Current	
		Full Step (Total)	Microstep RMS (Peak)
4 wire	Parallel	4.0A (8.0A)	4.0A (5.66A)
4 wire	Series	2A (4.0A)	2.0A (2.8A)
6 wire	Unipolar	4A per phase	NA
6 wire	Series	2.8A (5.6A)	2.8A (4A)
6 wire	50% Copper	4A (8A)	4A (1.7A)

- Peak Current= One phase on and the other phase off.
- Peak Current =1.414 times RMS.
- RMS= Current per phase with both phases driven on (full step).
- RMS Microstep (or full step)= Both phases operating at equal currents.
- RMS = .707 times peak current.
- Total = Entire motor current.

Set-up for Current Calibration

The following is the basic setup and diagram for 2 phase current measurement:

A. The Amp meter can be digital or (preferably) analog.
B. The bridge rectifier must be rated above the maximum expected voltage and current.
C. A small capacitor (filter) may be needed across the meter.
D. A single meter circuit can be used, but two meters will indicate proper operation.
E. Additional meter protection circuitry may be desired (not shown).



Current Set-up Techniques

There are several basic methods used in establishing the initial motor current settings. The method used depends on the product model.

The following is a matrix of AMS products with adjustable current and the recommended (initial) current set-up techniques:

Model	**Type**	**Adjustment**	**Method (See Below)**
MAX-410/420	Microstep	Programmable	A1
CMAX-410/810	Microstep	Programmable	A1
SAX/DAX	Full/Half Step	Programmable	A2
CCB-26	Microstep	Single Potentiometer	B
DCB-241	Half Step	Single Potentiometer	B
DCB-261	Microstep	Single Potentiometer	B
DR-4M/PS	Microstep	DIP Switch	B
CDR-4/8MPS	Microstep	DIP Switch	B
DCB-264	Microstep	Dual Potentiometer	C
DCB-612	Microstep	Dual Potentiometer	C
ALL			D

****************** WARNING **********************

LIVE CONNECTING/DISCONNECTING MOTORS WILL CAUSE DAMAGE THAT IS NOT COVERED BY WARRANTY.

General Procedure for all Methods

Assume a 2 amp bipolar motor (4 wire, parallel connection). The RMS value is 2 amps per phase, thus the peak (only one phase on) is 1.414 x 2 (amps), or 2.8 amps. Before proceeding, make sure the power is off and let any residual power supply capacitors discharge whenever motor circuits are connected or disconnected.

1. Adjust the output current to zero, either by pot adjustment, or serial command (depending on the product model/features).
2. Connect an amp meter(s) and motor as shown above.
3. Apply power.
4. Enable drive (method depends on model. See “E” command). The enable should eliminate “hold” reduction.
5. Increase the current setting until some amperage reading is obtained. Do not exceed the RMS current rating (2 amps in this example).
6. Adjust the “run” current. This is done at standstill. Methods for adjusting the current vary depending on the product model, as follows:

Method A: Programmable Current

AMS “programmable current” products have a digitally controlled potentiometer that is used for both hold and run current settings. The range is between 0 and 100 representing 0% and 100% of the full-scale drive current. Two “Y” command parameters control the hold and run values. For this procedure, set both values the same, i.e., “Y 40 40.” Generally the preferred method is use of the peak value (one phase maximum) for micro step models and RMS (both phases on) for full/half step models such as the SAX or DAX.

1. Microstep Models with Programmable Current:
   - 1A. Set the resolution mode to “fixed” resolution (H 0).
   - 1B. Single-step until a maximum current on one phase is reached.
   - 1C. Use the “Y” command to obtain the desired current.

2. Full/Half Step Models with Programmable Current:
   - 2A. Single-step until equal currents on both phases is reached.
   - 2B. Use the “Y” command to obtain the desired current.

Fine tune using the Empirical Method (D) as required.

Method B: Peak Current, Single Potentiometer Models

The single turn potentiometer’s position is proportional to the full current rating of the product. If necessary the driver is stepped until one phase is maximum and the other is at zero current (¼ step resolution is convenient).

1. Adjust the “run” current to the peak value, which is 2.8 amps in this example. Fine tune using the Empirical Method (D) as required.

Method C: RMS Current for Dual Potentiometer Models

The two motor windings are adjusted separately in these microstep designs. There is one Sine (SIN) and one Cosine (COS) control. The preferred method is to enable both outputs to produce equal currents (RMS), and then adjust both potentiometers to equate the values.

1. Adjust both potentiometers to minimum (CCW). After applying power, the “E 3” command presets the digital phase data to equal values corresponding to the RMS value. One potentiometer (either one) is then increased to the desired RMS current.
2. A comparing LED is provided that changes state when both current settings are equal. Once one phase current is set, the other potentiometer can be “tweaked” until the LED changes to the opposite, on or off, condition. Both meters will affirm equal currents.

Fine tune using the Empirical Method (D) as required.

Method D: Empirical Method, Minimum Current

The "empirical" method is the best approach for "final-tuning" the system and can/should be used for all AMS products. This technique is generally used for "final tuning" complete system configurations. When the best values are determined they can be used in future production, providing tolerances are sufficiently close. Once the system is assembled in its final form and the motion commands are sent to the motor:

1. Reduce the current by CCW rotation of the potentiometer(s) (by equal increments in dual potentiometer models, or by using the "Y" command available in programmable units).
2. Reduce the current setting until operation becomes erratic or undesirable.
3. Increase the current gradually until reliable operation is obtained, then increase the current equally by 10 to 20%.

For dual potentiometer models, both potentiometers must be adjusted by equal amounts. Note that the "E3" command must be issued if the motor has been stepped to a non-RMS position. Periodically use the "E3" sequence to balance the two (SIN/COS) currents.

In any of these adjustments, monitor motor temperature and insure that excessive heating does not occur. Larger motors require more time for temperature to stabilize. When a low hold current and short run cycle is used, heating effects are reduced.

HEAT: The Primary Enemy of Motor Damage

Advanced Micro Systems driver designs limit winding current to an adjustable value. Higher speeds are achieved from higher voltage DC supplies. In general, the only cause of motor damage is from excessive heating. Most step motors can withstand 100 degrees C.

A chopper drive regulates the motor current. Generally the "run" current is set at zero speed. If a hold mode is available the current will "set-back" when the motor is not moving. The voltage supplying the motor should be three or more times the rated motor voltage. If the supply were equal to the motor voltage, chopping would not function and performance would be very poor.

With higher voltages the regulation limits power and, hence heating. As the motor is rotated faster the chopper uses the available voltage to overcome a "back EMF" effect that takes place, thereby retaining more shaft torque.

More voltage applied to the motor equals more watts into the motor, resulting in additional heating. Motor heating can accumulate at continuous high stepping speeds, load torque and supply voltages of over 40 volts. Methods of reducing heating include motor heat sinking and short duty cycle (with a reduced holding current).