STEP1 and STEP2 Stepper Motor Control Modules

Control of a stepper motor requires both STEP1 and STEP2 modules. This documentation treats the STEP1 and STEP2 modules as a unit.

The STEP1 and STEP2 modules (see Figures 1 and 2) enable the IBM PC version of the Series 500/Soft500 package to independently control up to eight stepper motors. The STEP modules can set motor speed and direction and perform absolute and relative positioning of the motor shaft. A single STEP1 communicates with up to four STEP2's, each of which connects to a single external stepper motor drive. The motor drive translates the STEP2's control signals into the proper phase codes and wattage for the selected motor.

The STEP2 module is compatible with a wide variety of motor drives requiring TTL-level pulse and direction control signals. The STEP2 provides low-true and high-true outputs for pulse (STEP and STEP) and direction control (CW/CCW and CW/CCW). A complete motor control system can be assembled using a STEP1 and STEP2, a TTL-compatible motor drive, and a stepper motor compatible with the drive.

A single Series 500 can control a total of eight stepper motors if stepper control is the only function. For such applications, the system does not require analog input or A/D converter modules. It can, therefore, hold two STEP1 modules and eight STEP2 modules.

The STEP1 module contains a separate command buffer for each STEP2. A computer can issue several commands to a stepper motor almost instantaneously. However, the commands themselves may take seconds or minutes to be completed. Each command buffer receives all the pending commands for its associated motor, and passes the commands to the motor sequentially.

The STEP1 module contains a status register for each of its corresponding STEP2 modules. These registers enable a program to read the status of each motor. Status flags include "motor ready for command", direction, positioning mode, limit status, "motion complete", and motor ID.

Each STEP2 has a TTL-level LIMIT input which, when taken to logic low, halts the corresponding motor immediately. Execution of the limit condition is totally controlled by the STEP2 hardware. Triggering the limit input updates the limit flag in the status register and disables the motor. The motor will remain disabled until the control program resets the status register limit flag to "NO LIMIT". The STEP module instruction sets include commands for this purpose.

The LIMIT input responds to a switch closure to ground, or to a low-going edge from a TTL-level signal. The STEP2 also enters a limit condition if the system is powered up with a limit sensor already tripped. Once a limit condition has been set, the LIMIT input will only be affected by another switch closure or low-going TTL level. You may reenable and operate the the motor via software commands, even if the LIMIT input remains at logic low. The LIMIT input must recycle to provide another switch closure or low-going TTL edge to initiate another limit condition.

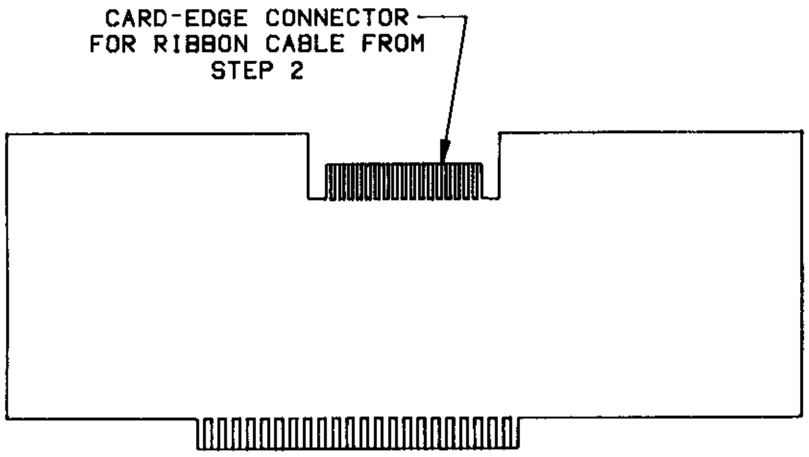


Figure 1. STEP1 Module

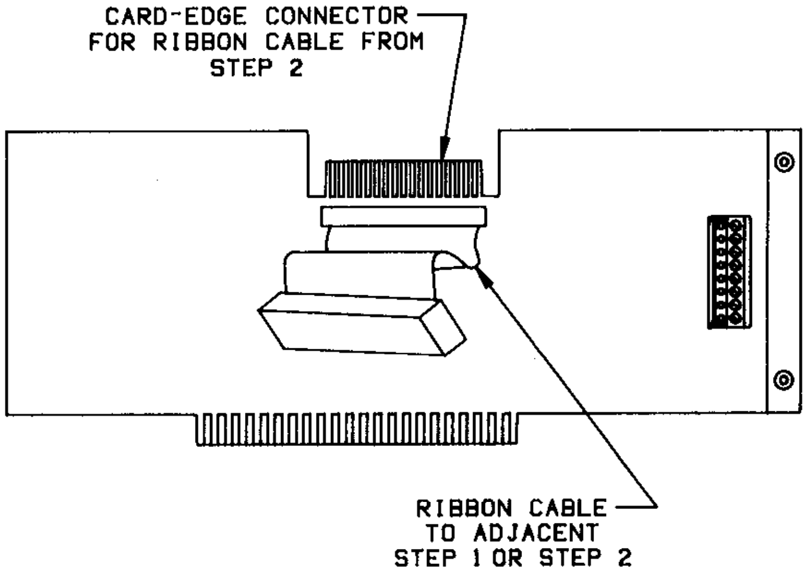

CARD-EDGE CONNECTOR FOR RIBBON CABLE FROM STEP 2
RIBBON CABLE TO ADJACENT STEP 1 OR STEP 2

Figure 2. STEP2 Module

User-Configured Components

The STEP1 and STEP2 modules have no jumpers or any other user-configurable components. However, the STEP2 module must be connected to a motor drive which itself may require configuration. Therefore, read the motor drive documentation thoroughly, and configure the motor drive before connecting it to the STEP2 module.

Motor control systems may require protection devices such as resistors or diodes for suppression of voltage surges or spikes from the stepper motor. Normally, the motor drive circuitry contains these devices; however, they may also be user-installed. Consult the motor drive documentation for specific instructions on any required protection devices.

Connections and Installation

CAUTION: Always turn off the system before installing or removing modules. After installing or removing modules, always replace the top cover and secure it with the screws. To reduce the possibility of EMI radiation, never operate the system with the top cover removed.

The outputs of the STEP2 module are compatible with motor drives having LS-TTL-compatible control inputs. A logic high level from a STEP2 output is a minimum of 2.7V with a maximum current drive capability of 0.4mA. Each STEP2 output can sink a maximum current of 4mA from the control inputs of the motor drive. For reliable operation, the voltage measured at a given control input of the motor drive should not exceed 0.4V when pulled to a logic low state by the STEP2.

Remove the output terminal block from each of the STEP2 modules. These are quick-disconnect terminals. A terminal block can be removed by pulling it from the STEP2 module in a perpendicular direction with a firm, even pressure.

Consult the motor drive documentation as to whether the motor drive requires high-true or low-true drive signals. Connect the STEP2's DIGITAL GROUND and appropriate STEP and CW/CCW output terminals to the motor drive.

Connect the LIMIT input terminal to any limit sensors required for the application. Limit switches or sensors are not a requirement for using the STEP modules. If the application uses more than one limit sensor, connect all limit sensors in parallel across the LIMIT input and DIGITAL GROUND.

When you have completed the connections, reinstall the terminal block on the STEP2 module. Repeat this operation for each STEP2.

The STEP modules must be installed in the Series 500 mother board contiguously, and in a specific order. You will need a number of consecutive slots equal to the total number of STEP1 and STEP2 modules. If necessary, rearrange the other modules in the Series 500. (Be sure to reassign slot designations in the configuration table if existing cards in the Series 500 are moved. You must also update the old slot designations in existing programs to the new slot designations.)

Plug STEP1 into the highest-numbered slot you have set aside. Plug the STEP2's in the next lower-numbered slots (see Figure 4). For a STEP1 plugged into slot *n*, the first STEP2 must be plugged into slot *n-1*, the second STEP2 into slot *n-2*, etc. A typical application might have a STEP1 in slot 8, and STEP2's in slots 7, 6, 5, and 4. The STEP2 closest to the STEP1 will control motor A, the next STEP2 will control motor B, etc.

Each STEP2 has a short ribbon cable and socket which fits a card-edge connector on top of the STEP1 or STEP2 module in the next higher slot. The ribbon cables and physical position of each STEP2 automatically configure the STEP2 for its associated motor. (You must also enter the module types and placement into the Soft500 configuration table.)

Connect the ribbon cable on the first STEP2 to the card-edge connector on top of the STEP1. Connect the ribbon cable on the second STEP2 to the card-edge connector on top of the first STEP2. Repeat this operation for the remaining STEP2 modules installed in the system.

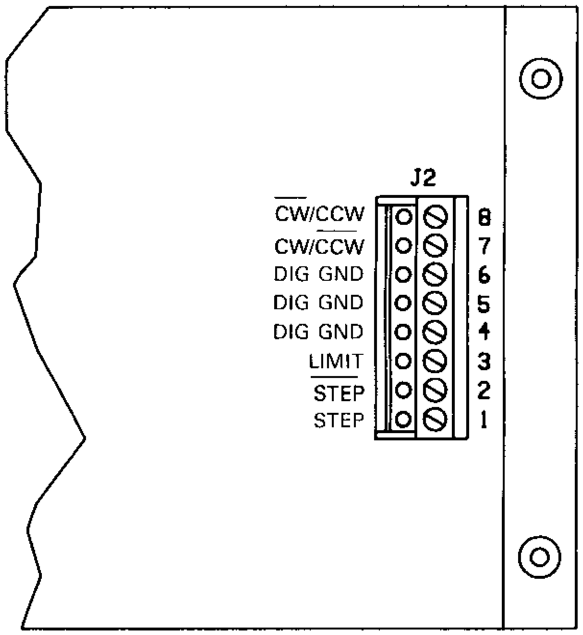

J2
$\overline{\text{CW}}$/CCW 8
CW/$\overline{\text{CCW}}$ 7
DIG GND 6
DIG GND 5
DIG GND 4
LIMIT 3
$\overline{\text{STEP}}$ 2
STEP 1

Figure 3. STEP2 Input and Output Terminals

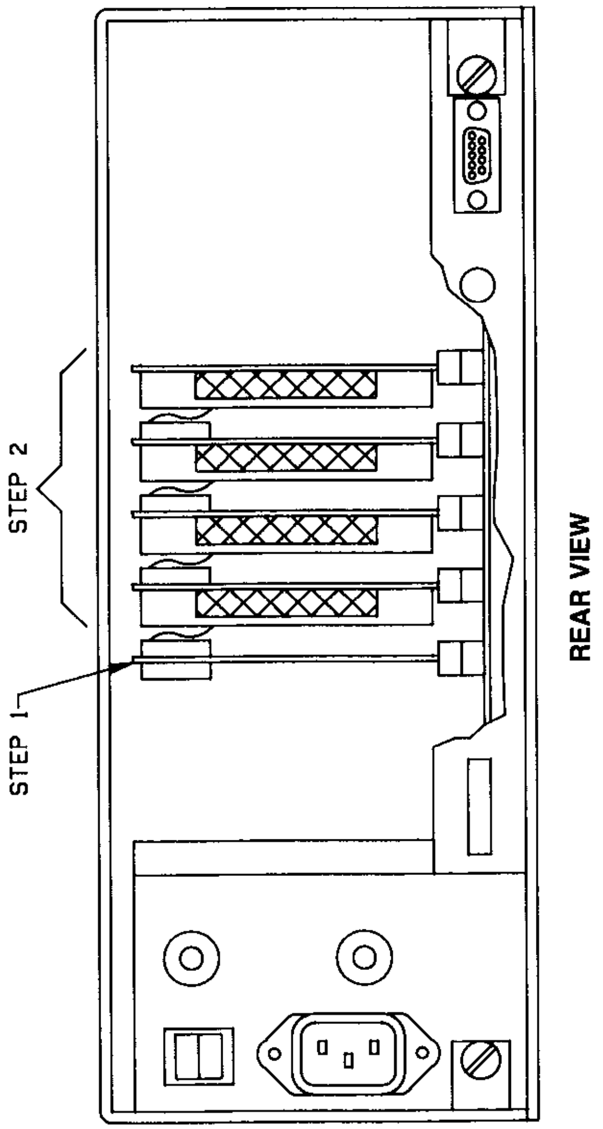


Figure 4. Installation of STEP1 and STEP2 in Series 500

STEP1 and STEP2 Programming Commands

The STEP1 and STEP2 modules are supported by Soft500 Version 4.0 or higher. Eight high-level commands control the operating parameters for the stepper motors. The Soft 500 software manual describes these high-level commands.

You can also program the STEP1 and STEP2 modules directly using BASIC's PEEK and POKE statements, or the corresponding memory read and memory write statements of other languages. Fourteen commands can be written to the STEP1 module in this manner. This technique is useful for running the STEP1 and STEP2 modules with older versions of Soft500, or for controlling STEP modules with programs which do not run under Soft500.

When you program a STEP module set with PEEKs and POKEs, you must address all commands to the STEP1 module. This applies regardless of which STEP2 the command is intended for. In contrast, some Soft500 commands *are* sent to STEP2's according to the command function and the designated motor.

Each slot in the Series 500 mother board is assigned two command locations: CMDA and CMDB. CMDA and CMDB are slot-specific memory addresses which provide for communication with a module plugged into a given slot. Table 1 shows the addresses CMDA and CMDB for the Series 500 slots 1 through 10. The computer can check the status of the motors by reading address CMDA for STEP1. It can send commands to the STEP module set by writing data to address CMDB for the STEP1.

Table 1. CMDA and CMDB Addresses Corresponding to Series 500 Slots 1-10

Slot	CMDA	CMDB
1	CFF80	CFF81
2	CFF82	CFF83
3	CFF84	CFF85
4	CFF86	CFF87
5	CFF88	CFF89
6	CFF8A	CFF8B
7	CFF8C	CFF8D
8	CFF8E	CFF8F
9	CFF90	CFF91
10	CFF92	CFF93

(If you reset the interface card to another base memory address, you must change the first three characters of the address (CFF). For instance, if you relocate the interface to address segment AFF0, the addresses in the table would range from AFF80 to AFF93.)

Low-Level Stepper Commands Addressed Through CMDB

The following low-level commands enable you to program the STEP modules via PEEKS and POKES to address CMDB for the STEP1. You may POKE values in either decimal or hexadecimal format. Observe the proper syntax for poking values as hexadecimal numbers: each value must begin with "&H" or "&h". Consult the computer's BASIC manual for more information on PEEKs, POKEs, and numerical formats.

It is only necessary to reprogram a particular operating mode or parameter if its current status is not the desired status. At power up the default states for the STEP modules are as follows:

Positioning Mode = Relative
Direction = CW
Continuous Speed = 0
Maximum Ramp Speed = 4096 sps (Range 0)
Motor Status Register Selected = A

Some commands written to CMDB require parameters. Any parameters accompanying a STEP command must be two bytes long regardless of the number of bytes needed to represent the parameter. Therefore, you must transfer a total of three bytes to CMDB for these types of commands: one byte for the command and two bytes for the parameter. The bytes are sent in the order *command, high parameter byte,* and *low parameter byte.* For any parameter which is less than than 256(decimal) or 100(hex), the first byte sent will be 0.

You can calculate the contents of the high and low parameter bytes as follows:

HIGH BYTE = INTEGER (n / 256)
LOW BYTE = n - (HIGH BYTE x 256)

where n = the decimal equivalent of the parameter to be sent. BASIC's HEX$ function can be used to convert a decimal number directly to hexadecimal notation.

Table 2. STEP1/STEP2 Low-Level Command Set

Function	Command Byte (HEX) to CMDB — Motors A, B, C, D
SET RELATIVE POSITION MODE	**0, 1, 2, 3**

REMARKS: In relative positioning mode, motor can be commanded to move up to 65,535 steps CW or CCW. All moves are based on the present position, which is considered "0" in relative positioning mode.

The selected motor will remain in this mode until the mode is changed. It is only necessary to issue this command if the currently programmed mode is not the desired mode.

Move motor to new position with "MOVE TO POSITION" command. Motor will move in the last programmed direction.

Relative Position Mode is the default mode at power-up.

PARAMETERS: None

SET ABSOLUTE POSITION MODE	**8, 9, A, B**

REMARKS: In absolute positioning mode, motor may be commanded to move to a new step position 0 to 65,535.

Present position must first be set as a value from 0 to 65,535 with the "SET PRESENT POSITION TO A VALUE" command, or set as "HOME" (position 32,678) with the "SET PRESENT POSITION TO HOME" command.

The selected motor remains in this mode until the mode is changed. It is only necessary to issue this command if the currently programmed mode is not the desired mode.

Move motor to new position with "MOVE TO POSITION" command. If the new position is numerically higher than the present position, motion will be in a CW direction. If the new position is numerically lower than the present position, motion will be in a CCW direction. The STEP modules calculate the direction automatically when the motor is commanded to move.

PARAMETERS: None

SET CONTINUOUS SPEED **90, 91, 92, 93**

REMARKS: Motor will rotate in the direction last programmed at the speed designated by the accompanying parameter bytes. Issue direction only if the currently programmed direction is not the desired direction.

Parameters: 0 - 65,535 steps per second.

MOVE TO POSITION **98, 99, 9A, 9B**

REMARKS: If motor is in relative positioning mode, it will move in the last programmed direction to the position designated by the parameter bytes.

If the motor is in absolute positioning mode, it will move in whatever direction is necessary to relocate to the position designated by the parameter bytes.

Parameters: Position value of 0 - 65,535.

TAKE A SINGLE STEP **20, 21, 22, 23**

REMARKS: Motor will move one step in the programmed direction. Direction need only be programmed if the desired direction is not the currently programmed direction.

Parameters: None

SET CW DIRECTION **28, 29, 2A, 2B**

REMARKS: When a direction has been programmed, it will remain in effect until a new direction is programmed.

CW direction is the default direction at power-up.

Parameters: None

SET CCW DIRECTION **30, 31, 32, 33**

REMARKS: When a direction has been programmed, it will remain in effect until a new direction is programmed.

Parameters: None

SET PRESENT POSITION AS "HOME" 38, 39, 3A, 3B

REMARKS: Used only for absolute positioning. Sets the present position as the center position (32,768) in the absolute positioning mode's 65,535-step motion range.

PARAMETERS: None

RETURN TO "HOME" 40, 41, 42, 43

REMARKS: Used only in absolute positioning mode. Moves the motor to position 32,768.

PARAMETERS: None

SET MAXIMUM POSITIONING SPEED C8, C9, CA, CB

REMARKS: Limits the maximum rotational speed which a motor may achieve during a move. Maximum positioning speed does not affect maximum steady-state rotation speed as programmed with the "SET CONTINUOUS SPEED" command. Maximum positioning speed must be programmed at least once, and before any positioning commands are issued.

PARAMETERS: 1 - 16,000 steps per second

SET PRESENT POSITION TO A VALUE D0, D1, D2, D3

REMARKS: Used only in absolute positioning mode. Sets the present position of the motor to a selected position value in the absolute positioning mode's 65,535-step motion range.

PARAMETERS: position value of 0 - 65,535.

SELECT STATUS REGISTER 2C, 34, 3C, 44

REMARKS: Makes a given motor's status register available to the STEP1's output register. Status can then be read by PEEKing address CMDA of STEP1 (status of all motors connected to a module set are read through the STEP1).

PARAMETERS: None

STOP/PURGE UNEXECUTED COMMANDS 4C, 54, 5C, 64

REMARKS: Halts selected motor without ramp-down, resets limit indicator to "NO LIMIT", and purges unexecuted commands from motor's command buffer.

If motor has tripped a limit, this command reenables the motor, resets the limit indicator, and purges the command buffer.

PARAMETERS: None

SET RAMP RATE **94 (SETS ALL MOTORS)**

REMARKS: Sets the ramp rate in steps per second squared (spss) for all motors controlled by a given STEP1. This parameter must be set at least once, before any positioning commands are issued, to match the dynamics of the motor. Default is 4096 spss (range 0).

PARAMETERS: Two bytes must be sent. First byte will always be 0. Second byte will be 0 - 14.

0=4096 spss
1=4369 spss
2=4681 spss
3=5041 spss
4=5461 spss
5=5957 spss
6=6553 spss
7=7281 spss
8=8192 spss
9=9362 spss
10=10922 spss
11=13107 spss
12=16384 spss
13=21845 spss
14=32768 spss

Stepper Status Accessed Through CMDA

The status of a motor can be read back from address CMDA of the STEP1 module. Operational status flags include "motor ready for commands", limit condition, positioning mode, direction, and "motion complete". The motor status register also provides motor ID.

In order to read a motor's status, you must first assign the motor's status register to the STEP1's output register. POKE the appropriate SELECT STATUS REGISTER command byte to address CMDB of the STEP1. (See the previous section on commands addressed through CMDB). Once you assign a motor's status register to the STEP1's output register, it remains the active register until you select another motor's status register.

Read the status of a motor by PEEKing the value at address CMDA of the STEP1. The status register updates automatically as the motor status changes, and can be read at any time.

You must logically "AND" the value PEEKed from CMDA with the decimal values 16, 8, 4, 2, or 1 to determine the operating status of the motor. The motor ID can be determined by doing a simple division and integer function on the status byte. The following table shows how to determine the motor status and ID:

Table 3. Stepper Motor Status as Read From CMDA

MOTOR READY: If ((PEEK (CMDA)) AND 16) = 0, then motor is not ready for command, else motor is ready.

LIMIT SWITCH: If ((PEEK (CMDA)) AND 8) = 0, then limit switch has not been tripped, else limit is tripped.

MODE: If ((PEEK (CMDA)) AND 4) = 0, then motor is in absolute positioning mode, else motor is in relative positioning mode.

DIRECTION: If ((PEEK (CMDA)) AND 2) = 0, then rotation is CCW, else motor rotation is CW.

MOTION COMPLETE: If ((PEEK (CMDA)) AND 1) = 0, then motor motion not complete, else motion is complete.

ID = MOTOR A: If INT (PEEK (CMDA) / 64) = 0, then status register is for Motor A.

ID = MOTOR B: If INT (PEEK (CMDA) / 64) = 1, then status register is for Motor B.

ID = MOTOR C: If INT (PEEK (CMDA) / 64) = 2, then status register is for Motor C.

ID = MOTOR D: If INT (PEEK (CMDA) / 64) = 3, then status register is for Motor D.

Service Information

Correct system operation relies not only on the STEP1 and STEP2 modules, but also on the motor drive and stepper motor. As a first step in troubleshooting a malfunctioning stepper control system, make sure the motor drive and motor are properly connected, and in good operating condition.

The STEP1 and STEP2 modules do not contain any user-serviceable components, calibrations screws, jumpers, or test points. The STEP module operating characteristics are controlled by firmware programmed into the STEP1's Read Only Memory. This can complicate hardware troubleshooting. Therefore, servicing the STEP modules is confined to diagnosing which module is faulty. Diagnosis consists of simple signal checking and substitution of a known good module for a suspected module.

Troubleshooting

First, check that all components in the stepper motor system are compatible. The STEP modules are intended for motor drives having LS-TTL-compatible control inputs. Connecting the STEP2 to other types of motor drives may cause problems.

A STEP2 module may be checked by testing its output levels with a DMM or oscilloscope. Use the low-level commands of Soft500 to command a motor to move at a speed of one step per second in a CW direction.

The STEP and STEP outputs should produce pulse trains consisting of square waves with a frequency of 1Hz. The outputs of STEP and STEP should be the inverse of each other.

When the motor is rotating in a clockwise direction, CW/CCW should be at a logic high level, while CW/CCW is low. Reversing the direction through software should exchange the levels at the CW/CCW and CW/CCW outputs.

Test the limit switch input of a STEP2 by manually shorting the input to ground. This should immediately stop the motor and set the limit flag in the motor's status register.

In systems which contain more than one STEP2, a problem with one motor may indicate a single faulty STEP2. If the suspected STEP2 and a known good STEP2 are exchanged and the problem follows the STEP2, the STEP2 is very likely faulty. If the problem remains with the same motor, either the motor drive, motor, or STEP1 are faulty.

STEP1 and STEP2 Specifications

Configuration: One STEP1 required to drive 1 to 4 STEP2 Modules

Channels: One motor channel per STEP2 module

STEP2 Outputs: LS TTL compatible, high or low true
  Output High: 0.4mA source @ 2.7V min.
  Output Low: 4mA sink @ 0.4V max.
  Output Signals: four; $\overline{\text{cw}}$/ccw, $\overline{\text{ccw}}$/cw, pulse and $\overline{\text{pulse}}$
  Pulse Signal Duration: 50% duty cycle

STEP2 Limit Input: LS TTL compatible, low true or switch closure to ground (internal pull-up resistor provided)
  Input High: 3.2V min.
  Input Low: 0.4mA @ 0.9V max.
  Signal Duration: ≥50ns

Slew Rate: 1 to 65,535 steps per second maximum in 1sps increments, software programmable

Speed Accuracy: 0.01% of full scale frequency

Positioning Modes:
  Absolute: 65,535 positions
  Relative: ±65,535 steps

Positioning Speed: 16,000sps maximum, software programmable

Ramp Rates: 15 rates, software selectable from 4,096sps^2 to 32,768^2

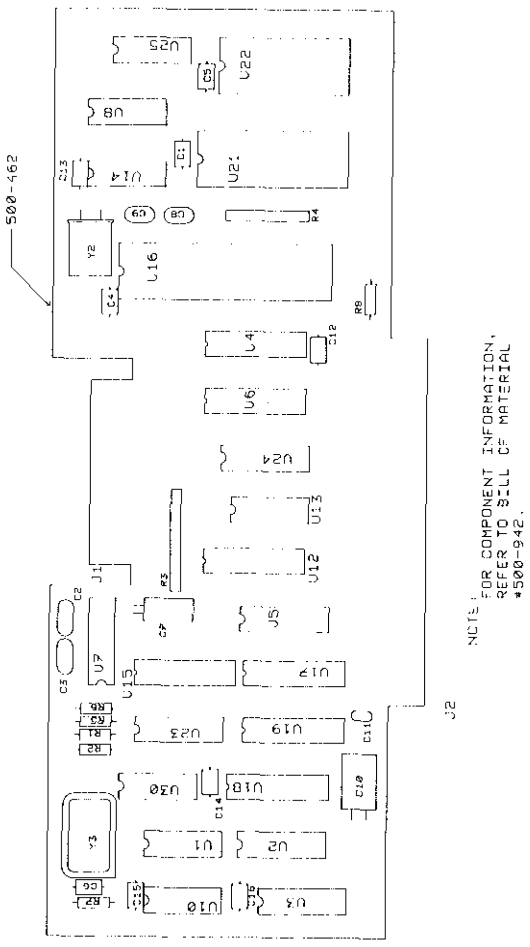
STEP1 on-board Microprocessor: 68B09 8/16 bit

Command Buffer Size: approx. 66 commands per motor (233 bytes)

Commands: 14 high level commands callable from BASIC and fully integrated with Keithley Data Acquisition and Control's Soft500 extended BASIC measurement and control software.

STEP1 PARTS LIST

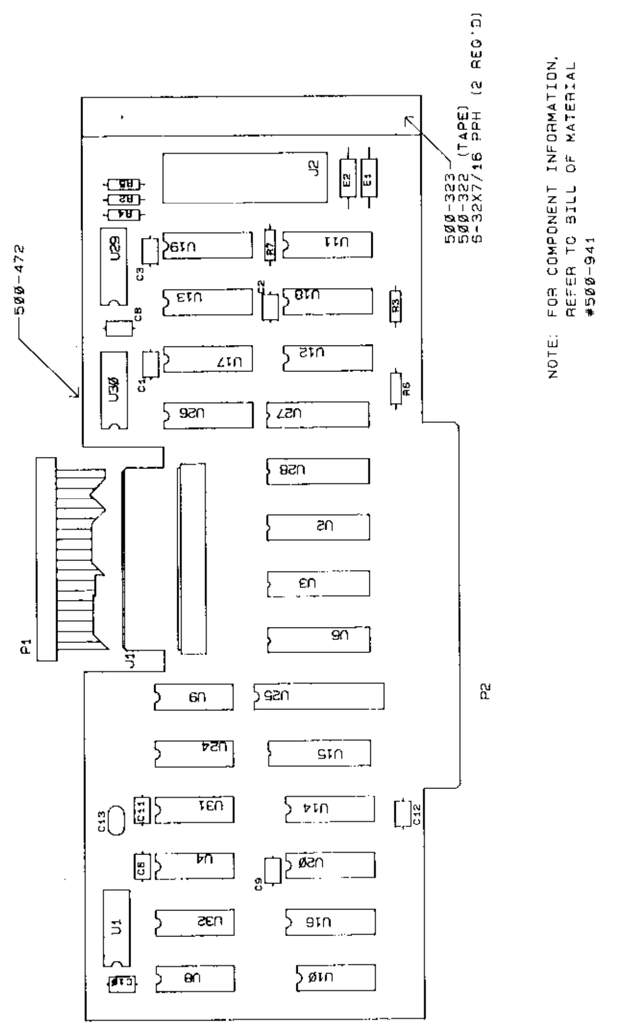
Part Number	Title	Remarks
C-22-22pF	Capacitor	C8, C9
C-237-.1	Capacitor	C11
C-237-1	Capacitor	C2, C3
C-314-22	Capacitor	C10
C-361-2.2	Capacitor	C7
C-365-.1	Capacitor	C1, C4, C6, C12, C16
CR-24-1	Crystal	Y2
CR-27	Crystal	Y3
IC-515	Int. Circuit (74HCT74)	U30
IC-337	Int. Circuit (74HC74)	U10
IC-338	Int. Circuit (74HC373)	U19
IC-351	Int. Circuit (74HC00)	U1
IC-354	Int. Circuit (74HC04)	U14
IC-397	Int. Circuit (74HCT374)	U17
IC-398	Int. Circuit (74HCT138)	U5, U24
IC-399	Int. Circuit (74HCT00)	U8
IC-444	Int. Circuit (74HCT04)	U25
IC-482	Int. Circuit (74HC377)	U18
IC-483	Int. Circuit (74HCT244)	U6
IC-486	Int. Circuit (74HC161)	U3, U23
IC-487	Int. Circuit (74HC379)	U2
IC-488	Int. Circuit (74HC688)	U15
IC-489	Int. Circuit (74HC244)	U12
IC-490	Int. Circuit (74HC20)	U13
IC-491	Int. Circuit (74HC245)	U4
IC-492	Int. Circuit (74HC123)	U7
500-800	Program	
500-801	Memory	
LSI-52	Int. Circuit (2764-20)	U21
MC-419	Label	
LSI-58	(6166)	U22
LSI-65	(MC68809)	U16
R-76-100k	Resistor	R1, R5, R6
R-76-10k	Resistor	R2, R7
R-263-1.96k	Resistor	R8
TF-179-1	Thick Film	R4
TF-180-1	Thick Film	R3
500-396	Shield, Component Side	
ST-143-1	Standoff	
500-397	Shield, Solder Side	
ST-137-2	Standoff	
500-398	Insulator, Solder Side Shield	
4-40×$^5/_8$ PPH	Screw, 4-40×$^5/_8$ Phil Pan Head	



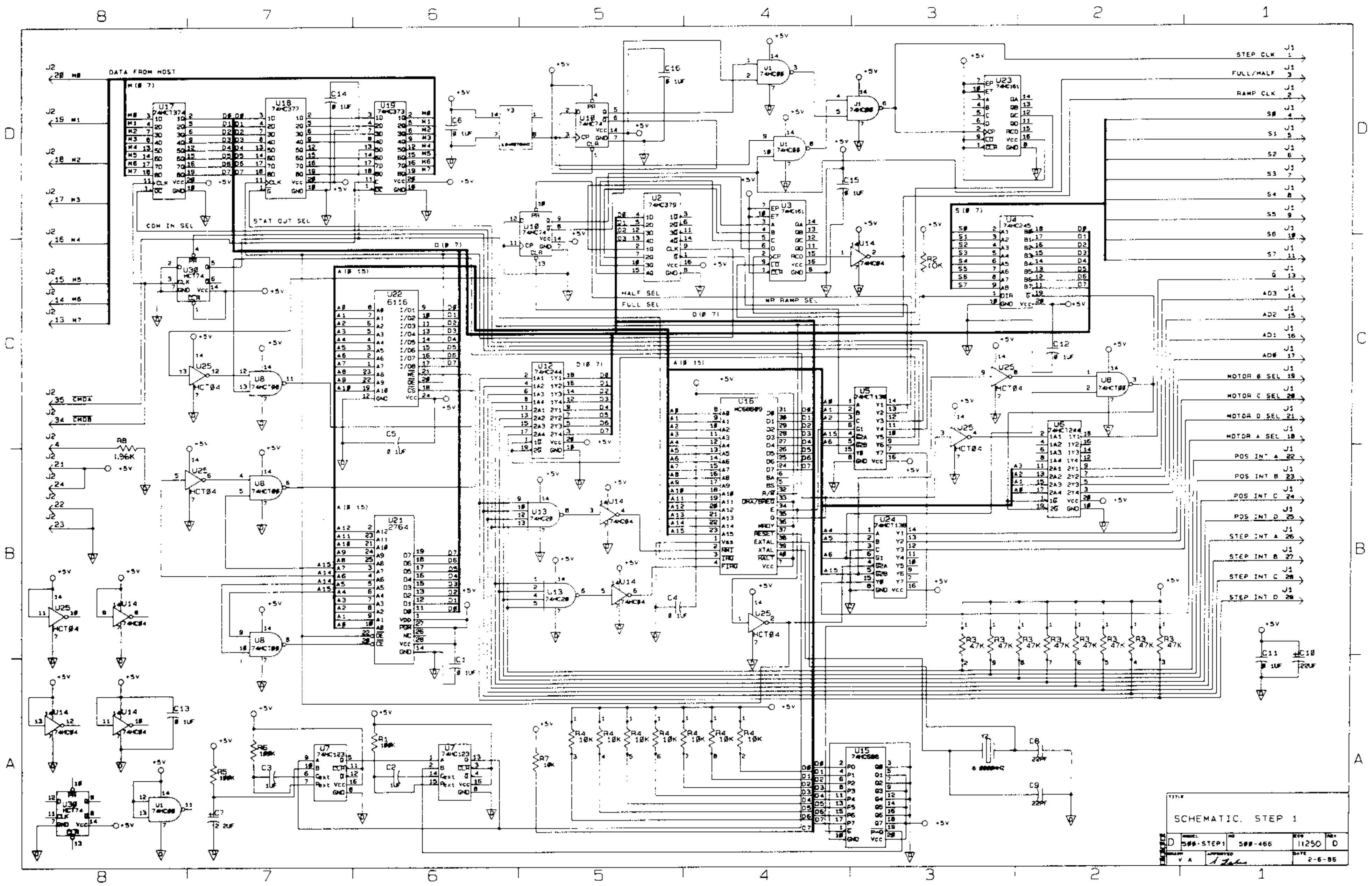
STEP1 COMPONENT LAYOUT

STEP2 PARTS LIST

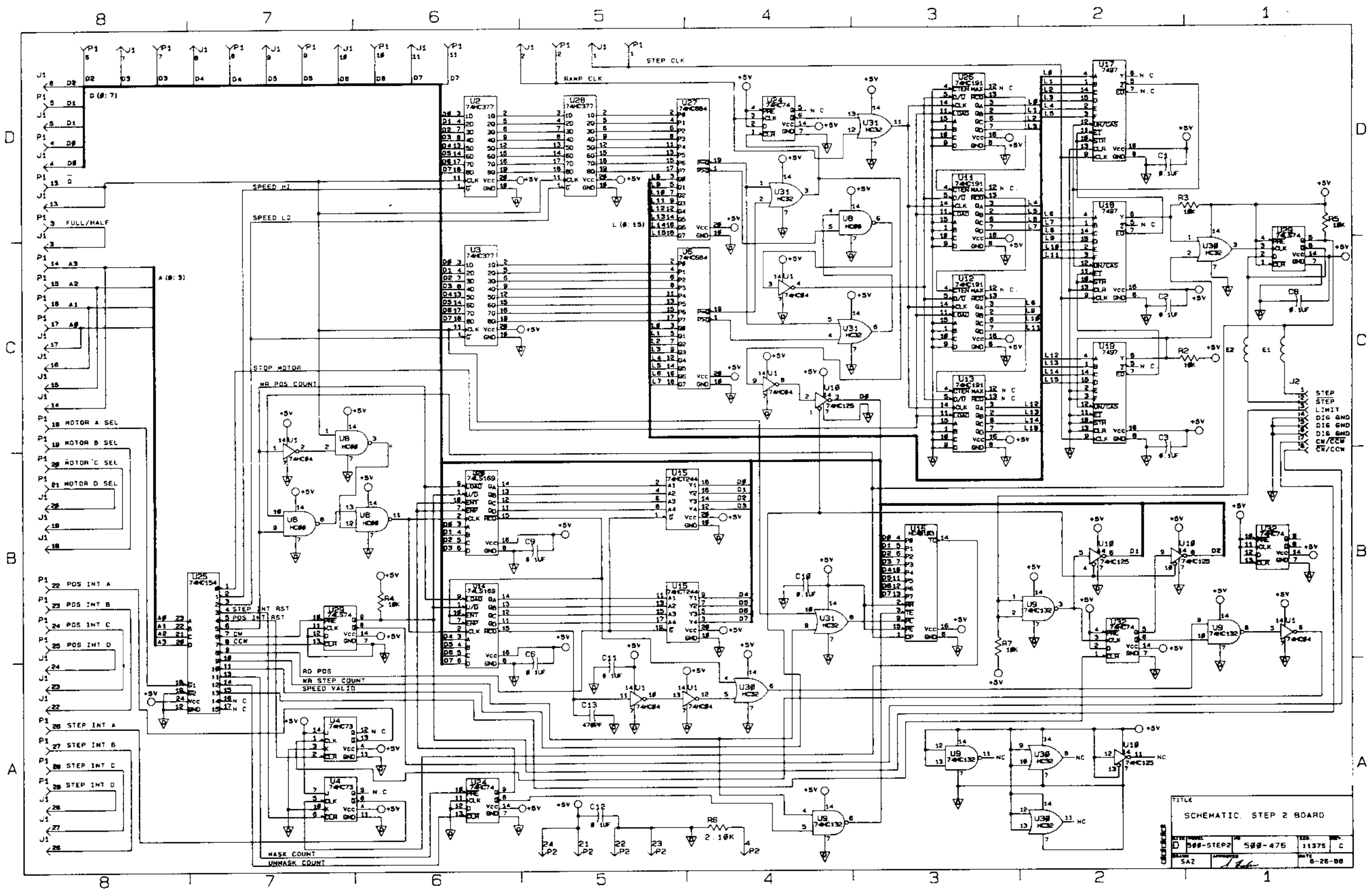
Part Number	Title	Remarks
C-365-.1	Capacitor	C1-C3, C6, C8-C12
CA-43-1	Cable Assembly	P1
CS-521-4	Connector	J2
CT-8	Ferrite Bead	E1, E2
IC-144	Int. Circuit (74LS74)	U29
IC-337	Int. Circuit (74HC74)	U24, U32
IC-351	Int. Circuit (74HC00)	U8
IC-354	Int. Circuit (74HC04)	U1
IC-451	Int. Circuit (74HC125)	U10
IC-475	Int. Circuit (7497)	U17-U19
IC-476	Int. Circuit (74HC154)	U25
IC-463	Int. Circuit (74HC32)	U30, U31
IC-477	Int. Circuit (74HC191)	U11-U13, U26
IC-478	Int. Circuit (74HC684)	U6, U27
IC-479	Int. Circuit (74HC73)	U4
IC-480	Int. Circuit (74HC40103)	U16
IC-481	Int. Circuit (74LS169)	U14, U20
IC-482	Int. Circuit (74HC377)	U2, U3, U28
IC-483	Int. Circuit (74HCT244)	U15
IC-513	Int. Circuit (74HC132)	U9
R-76-10k	Resistor	R2-R5, R7
R-263-2.10k	Resistor	R6
500-397	Shield, Solder Side	
ST-137-2	Standoff	
500-398	Insulator Solder Side Shield	
4-40×3/16 PPH	Screw, 4-40×3/16 Phil Pan Head	
500-323	Clamp Assembly	
500-322	Strip Rubber	
6-32×7/16 PPH		6-32×7/16 Phillips Pan Head Screw



STEP2 COMPONENT LAYOUT



STEP1 SCHEMATIC DIAGRAM



STEP2 SCHEMATIC DIAGRAM