Motion Engineering, Inc.

33 South La Patera Lane
Santa Barbara, CA 93117
ph (805) 681-3300
fax (805) 681-3311
info@motioneng.com

For the following MEI motion controllers:

CPCI Bus	*ISA Bus*	*PC-104 Bus*	*STD Bus*
CPCI/DSP	PCX/DSP	104/DSP	SERCOS/STD
PCI Bus	LC/DSP	104X/DSP	STD/DSP
PCI/DSP	SERCOS/DSP	SERCOS/104	*VME Bus*
			V6U/DSP

DSP Series
Motion Controller
Installation Guide

Mar 2002

DSP Series Motion Controller
Installation Guide

Mar 2002
Part # M001-0001 rev. B

Motion Engineering, Inc.
33 South La Patera Lane
Santa Barbara, CA 93117-3214
ph 805-681-3300
fax 805-681-3311
e-mail: technical@motioneng.com
website: http://www.motioneng.com   ftp site: ftp.motioneng.com

Motion Engineering, Inc.

CHAPTER 1 QUICK START

If you are familiar with motion controller connections, Quick Start offers a fast and easy installation procedure. If you are less familiar with motion controller connections, follow the procedures in Chapters, 2 - 6, which contain wiring diagrams and more detailed installation procedures.

If you have Windows NT, 95/98 or 3.11, then Motion Console is available for set-up procedures (the Microsoft Win32S extensions are available at no charge from Microsoft). Motion Console provides a powerful means to set-up, configure, test and debug motion control systems that use MEI controllers.

If you use only DOS, then see Appendix C, *SETUP.EXE* for set-up procedures.

For Servo Motors

1. Set the controller I/O address (0x300 is the default) using the on-board dip switches.

CPCI & PCI Users:	Because the CPCI and PCI controllers comply with the PCI Plug and Play specification, they do not have any on-board DIP switches. Instead, a software utility (included in your distribution) checks the address that the system assigns to the CPCI and PCI controllers. Refer to Section, *PCI on page 2-10*, for more information.

2. Install the controller in the computer.
3. Make sure the amplifier is turned off. Connect the encoders to the controller.
4. Install the MEI software as described in the release note included with the distribution. Run Motion Console (located in the *Motion Engineering* program group under *Start*).
5. In the *Hardware Summary* window, Click *Add Controller* (go to PCI tab if using PCI/DSP or CPCI controller).
   In the dialog box, enter a name for the controller.
   If a controller's address is different from the default, enter an address.
6. In the *Axis List*, double-click on an axis to open the *Axis Operation* window for that axis. Verify encoder operation by manually turning the motor shaft for the axis. As you turn the shaft, the *Actual* field in the *Position Status* display should change.
7. In the *Axis Operation* window, set the PID to zero by entering "0" in *Kp*, *Ki*, and *Kd* fields of the *Tuning Parameters* controls.
8. In the *Axis Operation* window, click the *Clear Position* and *Clear Fault* buttons.
9. Verify all motor and amplifier wiring, turn on and enable the amplifier. If the controller's amp enable output is connected to the amplifier, you must configure the amp enable logic (in the *Dedicated I/O* window). Next, to activate the amplifier, in the *Axis Operation* window click *Enable* in the *Amplifier* group.

   If the amplifier is in torque mode, you should be able to turn the motor shaft by hand.

   If the amplifier is in velocity mode, the motor shaft should be stiff. For more information, consult the amplifier manufacturer's documentation.

10. Verify the encoder phasing by entering positive and negative values in the *Offset* field of the *Tuning Parameters* display. Start at 10 and increase the offset until the motor is turning slowly. The *Actual* field in the *Position Status* section should display increasing values. If you enter a negative value in *Offset*, *Actual* should display decreasing values.
    If positive offset does not result in increasing encoder counts, then the encoder phasing is incorrect.
    Set the *Offset* to 0, turn off the amplifier and host computer, and exchange the A and B encoder leads. Repeat this procedure starting again with a 10 value in the *Offset* field to verify proper phasing.
11. Continue to exercise and tune the system as described in Appendix D, *Tuning Closed-Loop Systems*.

For Step Motors

1. Set the controller I/O address (0x300 is the default) using dip switches.

CPCI & PCI Users:	Because the CPCI and PCI controllers comply with the PCI Plug and Play specification, they do not have any on-board DIP switches. Instead, a software utility (included with your distribution) checks the address that the system assigns to the CPCI and PCI controllers. Refer to Section, *PCI on page 2-10*, for more information.

2. Install the controller in the computer and connect the step drive.
3. Make sure the step drive is turned off.
4. Install the MEI software as described in the release note included with the distribution. Start Motion Console (located in the *Motion Engineering* program group under *Start*).
5. Click *Add Controller* in the *Hardware Summary* window (go to PCI tab if using PCI/DSP or CPCI/DSP controller). Enter the name of the controller in the dialog box. Motion Console uses the default address 0x300 for the controller.
6. Choose the axis in the *Axis List* and click the *Configure Axis* button. On the *Axis Configuration* property page, configure the axis as *Stepper, Open Loop, Unipolar*. Click *Close*. If using an encoder, choose *Close Loop* and follow the instructions for phasing encoders in “For Servo Motors” (begin with step 7).
   Note!
   The Step output rate defaults to Slow (0-20 kHz). For greater step output, choose Medium (0-80 kHz), Fast (0-325 kHz) or Superfast (0-550 kHz) in the Axis Configuration property page.
7. In the *Axis Operation* window, click the *Clear Position* and *Clear Fault* buttons.
8. Turn on the step drive. Verify all motor and drive wiring , turn on the drive, and enable the drive. If the controller’s amp enable output is connected to the drive, you must configure the amp enable logic in the *Dedicated I/O* window. Then click *Enable* in the *Amplifier* group in the *Axis Operation* window to activate the amplifier.

   If the drive is in torque mode, you should be able to turn the motor shaft by hand. If the drive is in velocity mode, the motor shaft should be stiff. For more information, consult the amplifier manufacturer’s documentation.

9. Command a trapezoidal motion by entering *position*, *velocity*, and *acceleration* values in the *Axis Operation* window.
10. Verify that the motor turns one rotation when the appropriate number of steps are commanded.

Motion Developer's Support Program

Motion Engineering takes technical support seriously. We want your system to work! To continue to provide the best possible applications support, we have created the Motion Developer's Support Program. Participation in the Motion Developer's Support Program is required in order to receive applications support. Contact MEI for additional information.

MEI's Motion Developer's Support Program ensures that your critical project will receive the utmost applications support for timely problem resolution and faster development.

The Motion Developer's Support Program includes:

- One year of 24 hour/day, 7-day/week application technical support by telephone, e-mail, and/or fax (weekends and holidays included)
- Priority access to application engineers with reponse in the same business day
- Updated Motion Developer's Kit - provided on CD-ROM. This includes MEI's DSP Series development tools, libraries, and sample code for Windows NT, Windows 95/98, and Windows 3.x with the current MEI features, functions, and bug fixes
- One year of software maintenance and updates for MDK software, tools, libraries, and sample applications code.

How to Contact Us

Support is available through our corporate office:

24-hour support	*(805) 681-3300*
Fax	*(805) 681-3311*
e-mail	*technical@motioneng.com*

Software Updates

MEI periodically releases new software/firmware versions. New features are implemented, performance enhanced and new applications developed. The latest firmware/software releases are available on our FTP site at **ftp://ftp.motioneng.com**. These files are password protected, please contact MEI for information.

The DSP controller has non-volatile memory space to store the firmware and configuration parameters. All of the DSP Series controllers are compatible with the latest firmware and software versions. Firmware can be easily downloaded to the controller with CONFIG.EXE.

Future Controller Purchases

Motion Engineering ships the DSP Series controllers with the latest software, firmware, and on-board programmable logic. When building multiple machines we recommend that you save a configured version of your firmware to a diskette. The next time that you build a machine, load the firmware (from diskette) to the DSP Series controller (use the CONFIG program). This method is easiest.

We are constantly adding new features and improving the capability of our controllers. The hardware and on-board programmable logic are revised to meet the increasing demands. All future hardware/programmable logic revisions are backwards compatible with older software/firmware revisions, and future new features can be enabled with the latest versions of software/firmware.

VERSION.EXE

The VERSION program reads the current firmware version number and hardware identity from the DSP Series controller and displays them on the screen. The firmware version and option numbers can be read directly from your application code with the functions **dsp_version(...)** and **dsp_option(...).**

Firmware Versions

MEI always ships the DSP Series controllers with the latest software and firmware. The firmware, software, and Motion Console all have a version check built into the code. If the library version is incompatible with the firmware version, controller status will be listed as “bad” in Motion Console’s *Controller List* and the controller will be inaccessible.

If you wish to use an earlier version of the firmware on a newly purchased controller, or if you have an older controller and want to use a new firmware version, run the CONFIG program as described in the section in *CONFIG.EXE Board Configuration Program*, or use the Motion Console application.

Note that current firmware versions are available 24 hours a day on Motion Engineering’s FTP site (*ftp.motioneng.com*). Files for downloading are located in the */pub* directory. These files are password protected, please contact MEI for information.

CHAPTER 2 CONFIGURE & INSTALL BOARD

Overview

Warning!	MEI motion controller boards are sensitive electronic devices and require handling with proper ESD protection. Please do not touch the controller's bus interface.

Basically, there are 4 steps to installing DSP Series controllers:

1. Set an I/O address for the controller that does not conflict with any peripheral devices.

CPCI & PCI Users:	Because CPCI & PCI controllers comply with the PCI Plug and Play specification, they do not have any on-board DIP switches. The system assigns the address and IRQ resources to each device at bootup. The software utility Motion Console, supplied with the distribution CD-ROM, returns the resources assigned to the controller.

2. Set the Interrupt Request Level (IRQ) for the controller (optional).
3. Install the controller in computer.
4. Verify communication using Motion Console. (SERCOS controllers must be initialized before verifying communications. See *DSP Series C Programming Reference* for more information).

Detailed instructions for each of these steps are organized by individual controllers.

STCs and Cables

We recommend using STC modules to provide quick and easy screw terminal connections to the controller’s signals. Basically, you connect the controller to the STC modules using ribbon cables, and then you connect the rest of the system to the STC modules (using discrete wires). STC's mount on standard DIN rail.

For the PCI, connect the controller to the STC modules using high-density shielded, twisted pair cables, and then you connect the rest of the system to the STC modules (using discrete wires). STC’s mount on standard DIN rail.

Using STCs with ribbon cables provides your system with a clean and reliable interface. All ribbon cables are tested at the factory.

For the PCX, CPCI, STD, 104X & V6U

STC MODULES FOR PCX, CPCI, STD, 104X, V6U

STC-20 - Connection module for analog inputs, 1 required per controller.

STC-26 - Connection module for motor axes, 1 required for 2 motor axes.

STC-50 - Connection module for I/O lines, 1 required for each I/O header.

CABLES FOR PCX, CPCI, STD,104X, V6U

CBL-20 - Analog input ribbon cable, 1 required per controller.

CBL-26 - Motor axes ribbon cable, 1 required for every 2 axes.

CBL-50 - I/O ribbon cable, 1 required for every I/O header.

For the 104 & LC

STC MODULE FOR 104, LC

STC-50 - Connection module for I/O lines, motor and encoder feedback. One required for every 2 axes.

OPTOCON FOR 104, LC

OptoCon - Optically isolated Screw Terminal Connection module. Provides optical isolation for dedicated and user I/O. The OptoCon is a pin compatible replacement for the STC-50. One required for every 2 axes.

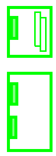
CABLES FOR 104, LC

CBL-50 - Ribbon cable.

CBL-100 - 100 pin high density male connector (Hirose) that mates to a 100 pin high density female (Hirose, #HIF6100-1.27R) connector and breaks out into 2 standard 50-pin ribbon cables required for 104 and LC.

For the PCI

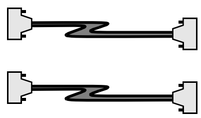
STC MODULES FOR PCI



STC-D50 - Connection module for User I/O, 1 required per controller.

STC-136 - Connection to four axes of I/O, 1 required for per controller.

CABLES FOR PCI

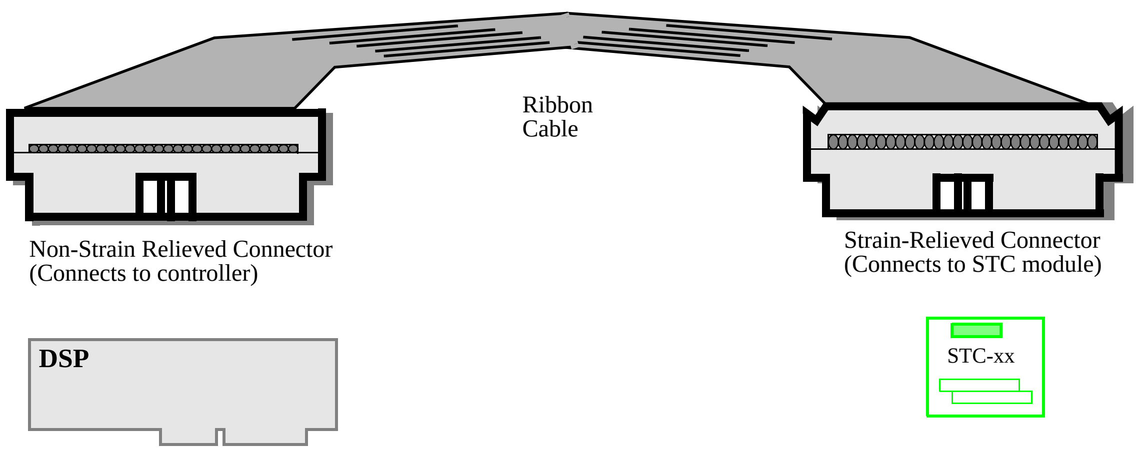


CBL-68 - Shielded cable for I/O connections, 2 required per controller.

CBL-50V - Shielded cable for User I/O connections, 1 required per controller

Cable Connectors

When installing MEI ribbon cables (ribbon cables are not used with the PCI controller), notice that the connectors (one at each end) are *different*. The non-strain relieved connectors fit into the headers on the controller. The strain relieved connectors fit into the STC modules.



CABLE CONNECTORS

I/O Port Address Space for PC-based Architectures

The DSP is mapped into the I/O space of the host CPU. The base I/O address is the first address of a 16 byte contiguous block of addresses. Starting with the base I/O address, the controller uses 16 address locations in the host computer's I/O space. All data transfers between the host computer and controller are done through this memory window.

Warning!	The controller must not share this I/O space (the 16 address locations) with any other devices.

The next table shows a typical mapping of I/O Port address space for PC-based architectures. (This does not include CompactPCI. See section *CPCI* on page 2-8).

Table 2-1 Typical Mapping of I/O Port Address Space

Hex Address	*Typical Uses*
200 - 20F	Game Control Adapter
210 - 237	*Not Used*
238 - 23F	Bus Mouse
240 - 277	*Not Used*
278 - 27F	Second Printer Port
280 - 2AF	*Not Used*
2B0 - 2DF	EGA
2E0 - 2E7	GPIB
2E8 - 2EF	Extra Serial Port
2F0 - 2F7	*Not Used*
2F8 - 2FF	Serial Port 2

Hex Address	*Typical Uses*
300 - 31F	Prototype Card
320 - 32F	XT Hard Disk
330 - 377	*Not Used*
378 - 37F	Printer Port
380 - 3AF	SDLC
3B0 - 3BF	Mono & Printer
3C0 - 3CF	EGA
3D0 - 3DF	CGA
3E0 - 3E7	*Not Used*
3E8 -3EF	Extra Serial Port
3F0 - 3F7	Disk Drive Controller
3F8 - 3FF	Serial Port 1

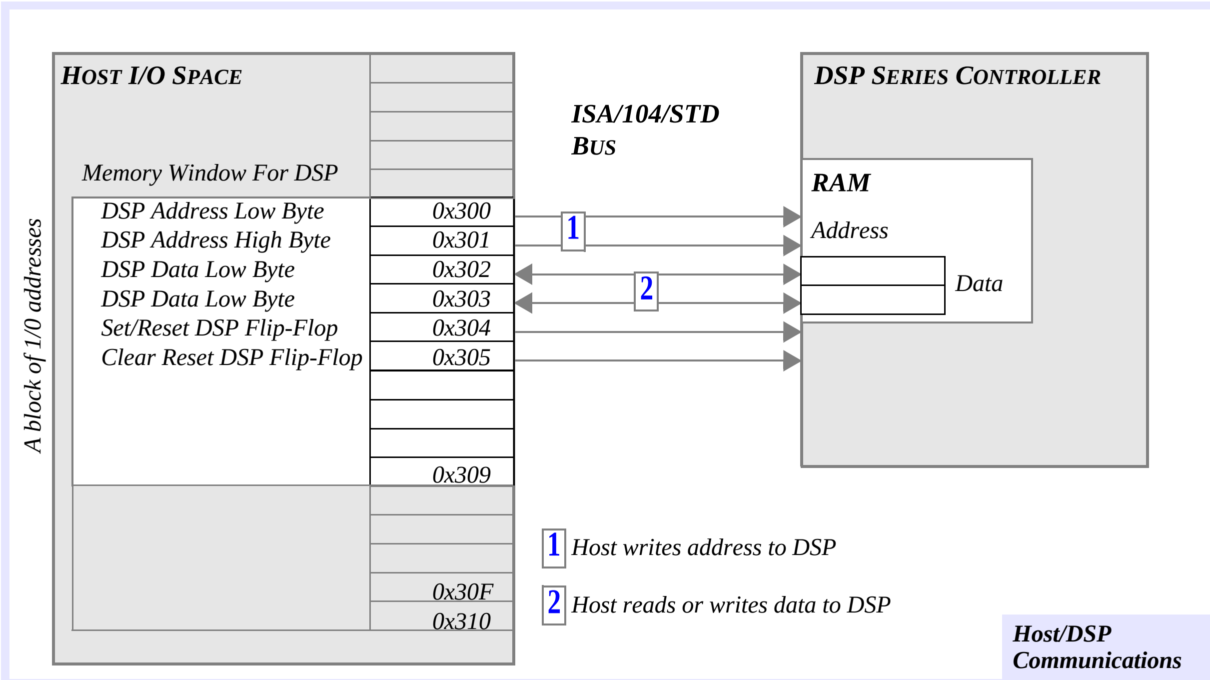
Base I/O Address Usage

Communication between the host CPU and the DSP Series controller occurs through a memory window. The start of this memory window is set by the address switch SW1 (for all DSP controllers except V6U). The DSP Series controllers use 6 addresses on the ISA/104/STD bus (see next table).

Table 2-2 6 Addresses on the ISA/104/STD bus

Address	*Description*	*Read/Write Size*
0x300	Address Low	8-bit Write Only
0x301	Address High	8-bit Write Only
0x302	Data Low	8-bit Read/Write
0x303	Data High	8-bit Read/Write
0x304	Set/Reset Flip-Flop	8-bit Write Only
0x305	(Clear Reset) Flip-Flop	8-bit Write Only

Figure 2-1 Host/DSP Communications over ISA/104/STD Bus



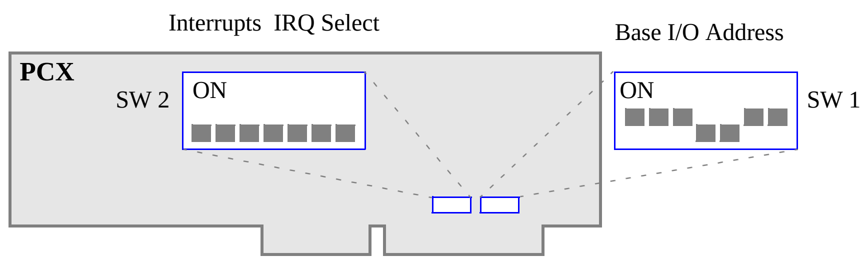
Communication occurs in 2 steps.

1. First the address is set by writing to 0x300 and 0x301 with two 8-bit writes. This "connects" the ISA bus data lines to the specified location in the controller's internal memory map.
2. Next, the data is read/write on addresses 0x302 and 0x303 with two 8-bit read/writes.

PCX

Locate Switches

Figure 2-2 PCX Address and IRQ Switch Locations



PCX SWITCHES

Set Base I/O Address

Use the SW1 dipswitch on each controller to set the base I/O address.

Table 2-3 Base Address Switch SW1

Address	***8***	***7***	***6***	***5***	***4***	***3***	***2***	***1***	
240	on	on	OFF	on	on	OFF	on	on	
250	on	on	OFF	on	on	OFF	on	OFF	
260	on	on	OFF	on	on	OFF	OFF	on	on = low
270	on	on	OFF	on	on	OFF	OFF	OFF	OFF = high
300	on	on	OFF	OFF	on	on	on	on	*Default*
310	on	on	OFF	OFF	on	on	on	OFF	
320	on	on	OFF	OFF	on	on	OFF	on	
330	on	on	OFF	OFF	on	on	OFF	OFF	
340	on	on	OFF	OFF	on	OFF	on	on	
350	on	on	OFF	OFF	on	OFF	on	OFF	
360	on	on	OFF	OFF	on	OFF	OFF	on	
370	on	on	OFF	OFF	on	OFF	OFF	OFF	

Set the Interrupts

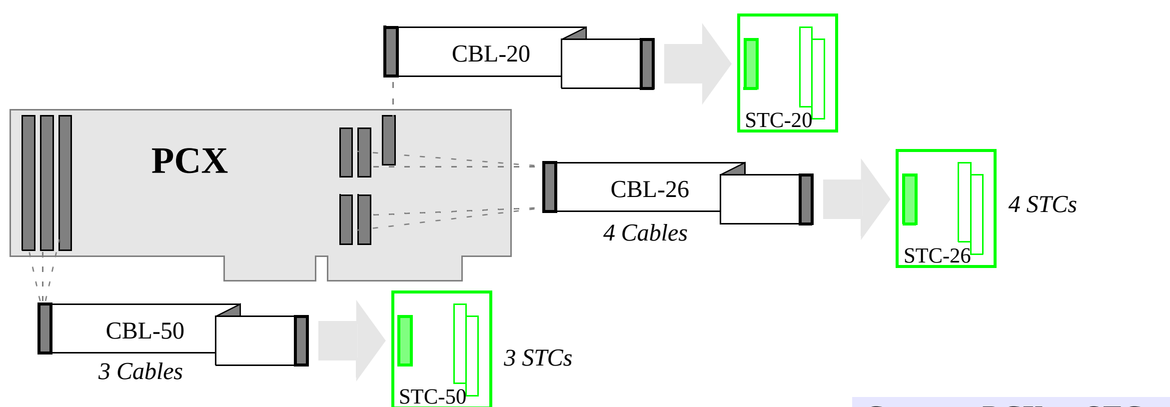
The DSP Series controllers can generate interrupts to the host CPU. SW2 connects the controller's interrupt circuitry to one of the host CPU's IRQ lines.

To use one of the IRQ lines, you must configure switch SW2. Configure switch SW2 for the interrupt, (IRQ2, IRQ3, ...) that you want the PCX to use.

Table 2-4 IRQ Switch SW2

IRQ	8	7	6	5	4	3	2	1	
None	off	off	off	off	off	off	off	off	*Default*
IRQ2	off	off	off	off	off	off	off	ON	
IRQ3	off	off	off	off	off	off	ON	off	
IRQ4	off	off	off	off	off	ON	off	off	
IRQ5	off	off	off	off	ON	off	off	off	
IRQ10	off	off	off	ON	off	off	off	off	
IRQ11	off	off	ON	off	off	off	off	off	
IRQ12	off	ON	off	off	off	off	off	off	
IRQ13	ON	off	off	off	off	off	off	off	

Connect Cables/Insert Board



Connect PCX to STCs

To install the controller:

1. **Turn off the power to your computer** and remove the cover.
2. Select any unused full-length expansion slot (16 or 32-bit) and remove its blank metal bracket from the computer.
3. Orient the controller inside the computer so that it lines up with the card-edge connector.
4. Press down on the metal bracket tab and the top of the board until the controller is firmly seated.
5. Feed cables through the back of the PC and connect the non-strained relieved connectors to the PCX.
6. Secure the bracket in place with the screw.
7. Proceed to Chapter 3 to test your I/O Address.

CPCI

No Switches

There are no switches on the CPCI. Because the CPCI complies with the PCI Plug-and-Play specification, the BIOS **automatically sets** the I/O addresses and IRQ of all peripherals in the system.

Accessing the CPCI

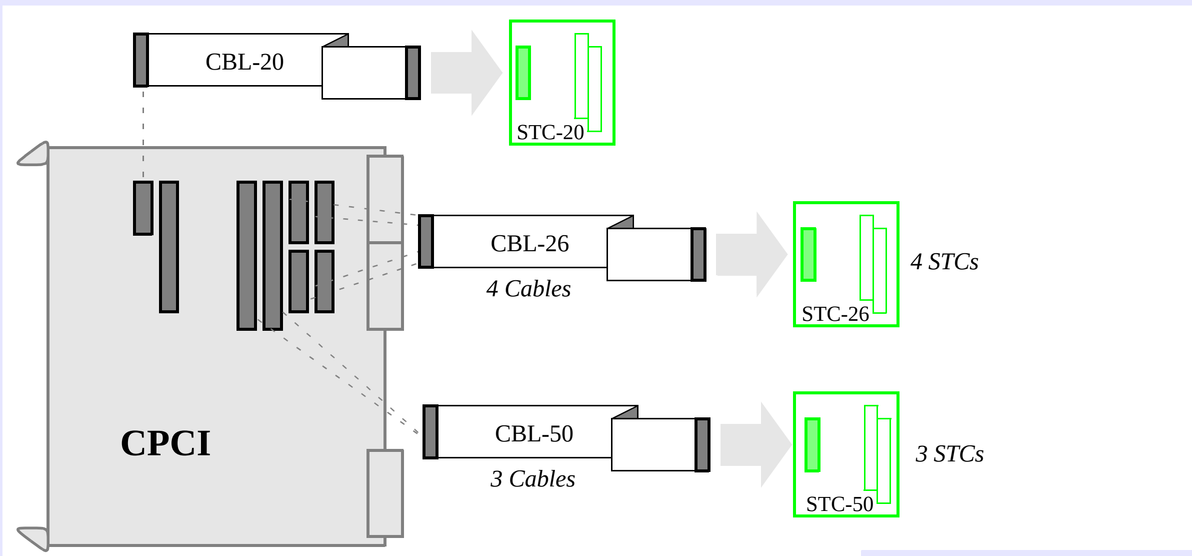
In order to properly access the controller using any MEI-supplied utility (Motion Console) or your own application program, you must obtain the address the BIOS gave to the CPCI-bus computer (at start-up). This can be determined by an MEI supplied function, *find_pci_dsp(...)*, or via Motion Console.

Motion Console will automatically find all PCI controllers on the bus. Simply select *Add Controller* and click on the *PCI Controller* tab. The address and IRQ of all PCI bus controllers will be listed.

Usage of the MEI function *find_pci_dsp(...)* is further described in the *C Programming Reference Manual.* Please refer to that document for a detailed description on using this function.

If you are still having problems communicating with the controller after you've found its address, you may have to reserve the resources in use by the controller using the *System* applet in the control panel.

Connect Cables/Insert Board



Connect CPCI to STCs

To install the controller:

1. **Turn off the power to your computer**.

2. Select an unused 6U expansion slot and remove its blank metal bracket from the computer.
3. Orient the controller so that it lines up with the card guides and insert the controller partially into the chasis.
4. Feed cables through the front panel and connect the non-strain relieved connectors to the CPCI.
5. Insert the controller fully into the slot until the injectors engage the chassis. Use the injectors to firmly seat the controller in the chassis.
6. Proceed to Chapter 3 to test your I/O Address.

PCI

No Switches

There are no switches on the PCI. The PCI controller supports PCI's Plug-and-Play addressing scheme, which means the BIOS **automatically sets** the addresses of all peripherals in the system.

Accessing the PCI

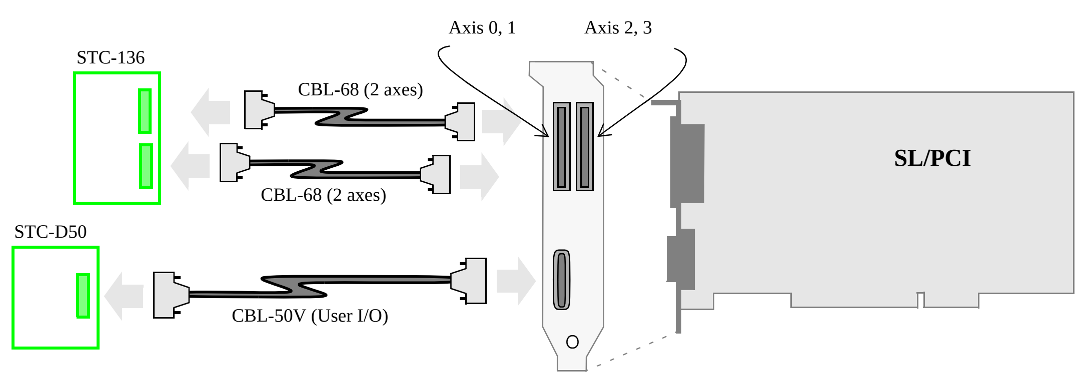
In order to properly access the controller using any MEI-supplied utility (Motion Console) or your own application program, you must obtain the address the BIOS gave to the PCI-bus controller (at start-up). This can be determined by an MEI supplied function, *find_pci_dsp(...)*, or via Motion Console.

Motion Console will automatically find all PCI controllers on the bus. Simply select *Add Controller* and click on the *PCI Controller* tab. The address and IRQ of all PCI bus controllers will be listed.

Usage of the MEI function *find_pci_dsp(...)* is further described in the *C Programming Reference Manual.* Please refer to that document for a detailed description on using this function.

If you are still having problems communicating with the controller after you've found its address, you may have to reserve the resources in use by the controller using the *System* applet in the control panel.

Connect Cables/ InsertBoard



Connect PCI to STCs

To install the controller

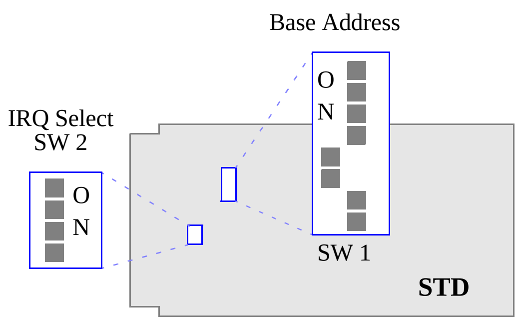
1. **Turn off the power to your computer**.
2. Select an unused expansion slot and remove its blank metal bracket from the computer.
3. Orient the controller so that it lines up with the card-edge connector.
4. Press down on the metal bracket tab and the top of the board until the connector is fully seated.
5. Secure the bracket in place with the screw.

Proceed to Chapter 3 to test your I/O Address

STD

Locate Switches

Figure 2-3 STD Address and IRQ Switch Locations



STD Switches

Set Base I/O Address

Use the SW1 dipswitch on each controller to set the base I/O address.

Table 2-5 Base Address Switch SW1

Address	***8***	***7***	***6***	***5***	***4***	***3***	***2***	***1***	
240	on	on	OFF	on	on	OFF	on	on	
250	on	on	OFF	on	on	OFF	on	OFF	
260	on	on	OFF	on	on	OFF	OFF	on	
270	on	on	OFF	on	on	OFF	OFF	OFF	
300	on	on	OFF	OFF	on	on	on	on	*Default*
310	on	on	OFF	OFF	on	on	on	OFF	
320	on	on	OFF	OFF	on	on	OFF	on	
330	on	on	OFF	OFF	on	on	OFF	OFF	
340	on	on	OFF	OFF	on	OFF	on	on	
350	on	on	OFF	OFF	on	OFF	on	OFF	
360	on	on	OFF	OFF	on	OFF	OFF	on	
370	on	on	OFF	OFF	on	OFF	OFF	OFF	

on = low
OFF = high

Set the Interrupts

Interrupts may be generated from the DSP Series controller to the host CPU. SW2 connects the controller's interrupt circuitry to one of the host CPU's IRQ lines.

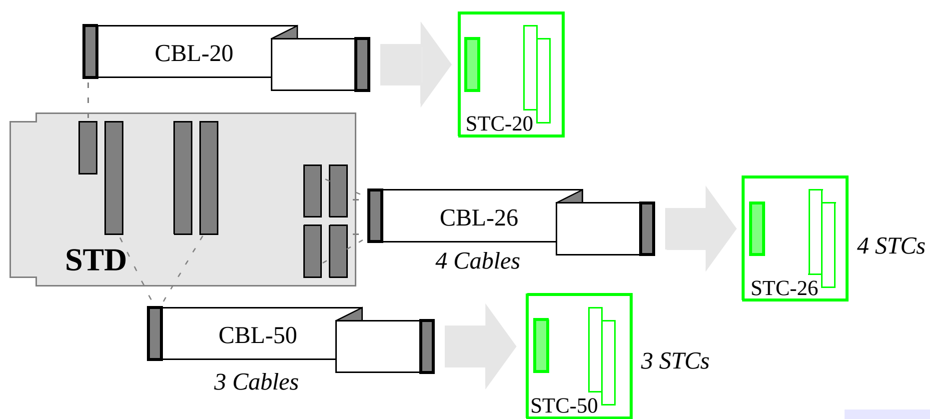
To use one of the IRQ lines, you must configure switch SW2. Configure switch SW2 for the interrupt (IRQ2, IRQ3, ...) that you want the STD to use.

Table 2-6 IRQ Switch SW2

IRQ	*4*	*3*	*2*	*1*	
None	off	off	off	off	*Default*
IRQX*	off	off	off	ON	
INTRQ3*	off	off	ON	off	
INTRQ	ON	off	off	off	
INTRQ1	off	ON	off	off	

**only supported by the STD-32 bus.*

Connect Cables/Insert Board



Connect STD to STCs

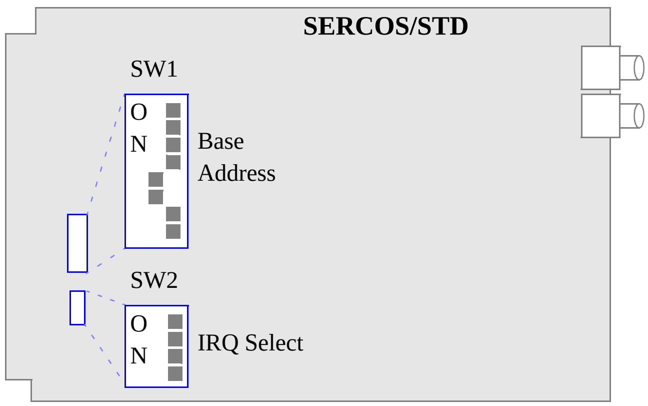
To install the controller:

1. **Turn off the power** to the STD card cage.
2. Select an unused slot (STD/32 or STD/80).
3. Install all required ribbon cables.
4. Insert the controller and press firmly until the board is seated in the card-edge connector.
5. Proceed to Chapter 3 to test your I/O Address.

SERCOS/STD

Locate Switches

Figure 2-4 SERCOS/STD Address and IRQ Switch Locations



SERCOS/STD Switches

Set Base I/O Address

Use the SW1 dipswitch on each controller to set the base I/O address.

Table 2-7 Base Address Switch SW1

Address	***8***	***7***	***6***	***5***	***4***	***3***	***2***	***1***	
240	on	on	OFF	on	on	OFF	on	on	
250	on	on	OFF	on	on	OFF	on	OFF	
260	on	on	OFF	on	on	OFF	OFF	on	
270	on	on	OFF	on	on	OFF	OFF	OFF	
300	on	on	OFF	OFF	on	on	on	on	*Default*
310	on	on	OFF	OFF	on	on	on	OFF	
320	on	on	OFF	OFF	on	on	OFF	on	
330	on	on	OFF	OFF	on	on	OFF	OFF	
340	on	on	OFF	OFF	on	OFF	on	on	
350	on	on	OFF	OFF	on	OFF	on	OFF	
360	on	on	OFF	OFF	on	OFF	OFF	on	
370	on	on	OFF	OFF	on	OFF	OFF	OFF	

on = low
OFF = high

Set the Interrupts

Interrupts may be generated from the DSP Series controller to the host CPU. SW2 connects the controller's interrupt circuitry to one of the host CPU's IRQ lines.

To use one of the IRQ lines, you must configure switch SW2. Configure switch SW2 for the interrupt (IRQ2, IRQ3, ...) that you want the SERCOS/STD to use.

Motion Engineering, Inc.

Table 2-8 IRQ Switch SW2

IRQ	***4***	***3***	***2***	***1***	
None	off	off	off	off	*Default*
IRQX*	off	off	off	ON	
INTRQ3*	off	off	ON	off	
INTRQ	ON	off	off	off	
INTRQ1	off	ON	off	off	

**only supported by the STD-32 bus*

Connect Cables/Insert Board

To install the controller:

1. **Turn off the power to the STD card cage** and remove the card clamp.
2. Select an unused slot (STD/32 or STD/80).
3. Insert the controller and press firmly until the board is seated in the card-edge connector.
4. Proceed to Chapter 3 to test your I/O Address, **then return to Step 4**.
5. Connect the fiber optic cables in a ring between the SERCOS/STD and the drives. The dark gray connectors are receivers ("Rx") and the light gray connectors are transmitters ("Tx"). Connect the controller's light gray connector to the first drive's dark gray (Rx) connector, the connect the first drive's light gray (Tx) connector to the second drive's dark gray (Rx) connector, etc.

   The light-emitting module on the controller can be turned on and off for testing with the functions **turn_on_sercos_led(…)** and **turn_off_sercos_led(…).** (See the *DSP Series C Programming Reference* for more information.)

6. When all drives are connected, turn on the power to the drives. Each drive begins an initialization sequence. Most drives have an LCD or LED display to indicate when the initialization is complete. Consult the specific drive documentation and chapter 6 *of DSP Series C Programming Reference* for more information about SERCOS initialization procedures.

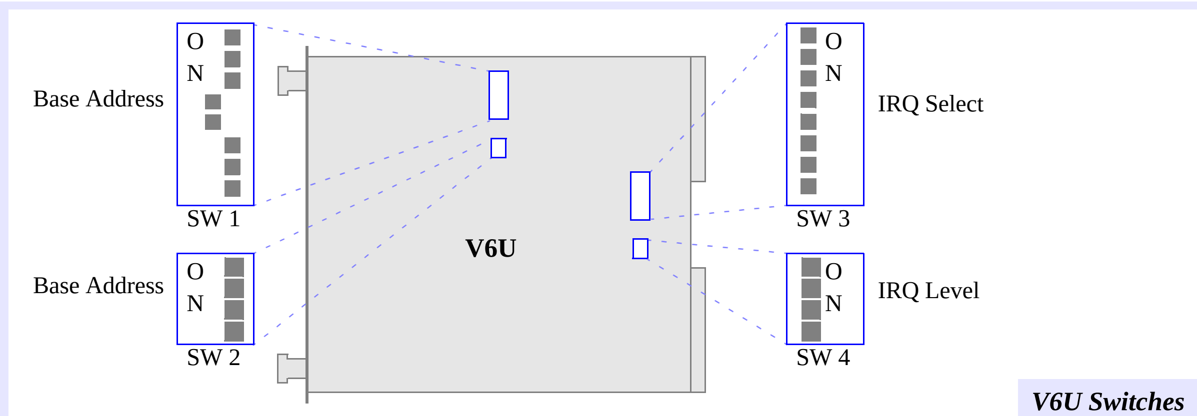
Once the SERCOS/STD has been initialized, you can exercise and tune the system using Motion Console.

V6U

Locate Switches

The base I/O address switch is located in the upper center of the V6U controller and is labeled SW1. SW2 is not currently used and should remain at its default setting (all ON). The IRQ Select and Level switches (SW3 and SW4) are located in the right mid-section of the controller.

Figure 2-5 V6U Address and IRQ Switch Locations



Set Base I/O Address

Use the SW1 and SW2 dipswitches on each controller to set the base I/O address. There are 10 possible choices for the Base I/O Address: 0xFFF0220, 0xFFF0240, 0xFFFF0260, 0xFFFF0280, 0xFFFF02A0, 0xFFFF0300, 0xFFFF0320, 0xFFFF0340, 0xFFFF0360, or 0xFFFF0380.

After choosing a Base I/O Address, look at the next 2 tables to find the switch settings that will implement your desired Base I/O Address.

Table 2-9 Base Address Switch (0xFFFF0220 - 0xFFFF02A0)

Bus	*Switch*	*0xFFFF0220*	*0xFFFF0240*	*0xFFFF0260*	*0xFFFF280*	*0xFFFF02A0*
A15	SW2-3	on	on	on	on	on
A14	SW2-2	on	on	on	on	on
A13	SW2-1	on	on	on	on	on
A12	SW1-8	on	on	on	on	on
A11	SW1-7	on	on	on	on	on
A10	SW1-6	on	on	on	on	on
A9	SW1-5	OFF	OFF	OFF	OFF	OFF
A8	SW1-4	on	on	on	on	on
A7	SW1-3	on	on	on	OFF	OFF
A6	SW1-2	on	OFF	OFF	on	on
A5	SW1-1	OFF	on	OFF	on	OFF

Table 2-10 Base Address Switch (0xFFFF0300 - 0xFFFF0380)

Bus	***Switch***	***0xFFFF0300***	***0xFFFF0320***	***0xFFFF0340***	***0xFFFF0360***	***0xFFFF0380***
A15	SW2-3	on	on	on	on	on
A14	SW2-2	on	on	on	on	on
A13	SW2-1	on	on	on	on	on
A12	SW1-8	on	on	on	on	on
A11	SW1-7	on	on	on	on	on
A10	SW1-6	on	on	on	on	on
A9	SW1-5	OFF	OFF	OFF	OFF	OFF
A8	SW1-4	OFF	OFF	OFF	OFF	OFF
A7	SW1-3	on	on	on	on	on
A6	SW1-2	on	on	OFF	OFF	on
A5	SW1-1	on	OFF	on	OFF	on
		Default				

The logic for the address switches are ON = low and OFF = high.

Communication between the host CPU and the DSP Series controller occurs through a memory window. The start of this memory window is set by the address switches SW1 and SW2. The DSP Series controllers use 6 addresses on the VME bus (see next table).

Table 2-11 Addresses on the VME bus

Address	***Description***	***Read/Write Size***
0xFFFF0300	Address Low	8 or 16-bit Write Only
0xFFFF0301	Address High	8-bit Write Only
0xFFFF0302	Data Low	8 or 16-bit Read/write
0xFFFF0303	Data High	8-bit Read/Write
0xFFFF0304	Set/Reset Flip-Flop	8 or 16-bit Write Only
0xFFFF0305	(Clear Reset) Flip-Flop	8-bit Write Only

Communication occurs in two steps.

1. First, set the address by writing to 0xFFFF0300 and 0xFFFF0301 with either two 8-bit writes, or a 16-bit write. This “connects” the VME bus data lines to the specified location in the controller’s internal memory map.
2. Next, the data is read/write on addresses 0xFFFF0302 and 0xFFFF0303 with either two 8-bit read/writes or a 16-bit read/write.

Set the Interrupts

The IRQ Select switch connects the V6U's interrupt circuitry to a particular IRQ line on the VME bus. To select a VME-bus IRQ line, turn ON the corresponding switch while leaving the other switches off.

For example, if IRQ3 is connected, SW3-3 must be ON while SW3-1, SW3-2, SW3-4, SW3-5, SW3-6, SW3-7, and SW3-8 must be OFF.

Table 2-12 IRQ Select Switch SW3

IRQ	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	
None	off	off	off	off	off	off	off	off	*Default*
IRQ1	off	off	off	off	off	off	off	ON	
IRQ2	off	off	off	off	off	off	ON	off	
IRQ3	off	off	off	off	off	ON	off	off	
IRQ4	off	off	off	off	ON	off	off	off	
IRQ5	off	off	off	ON	off	off	off	off	
IRQ6	off	off	ON	off	off	off	off	off	
IRQ7	off	ON	off	off	off	off	off	off	

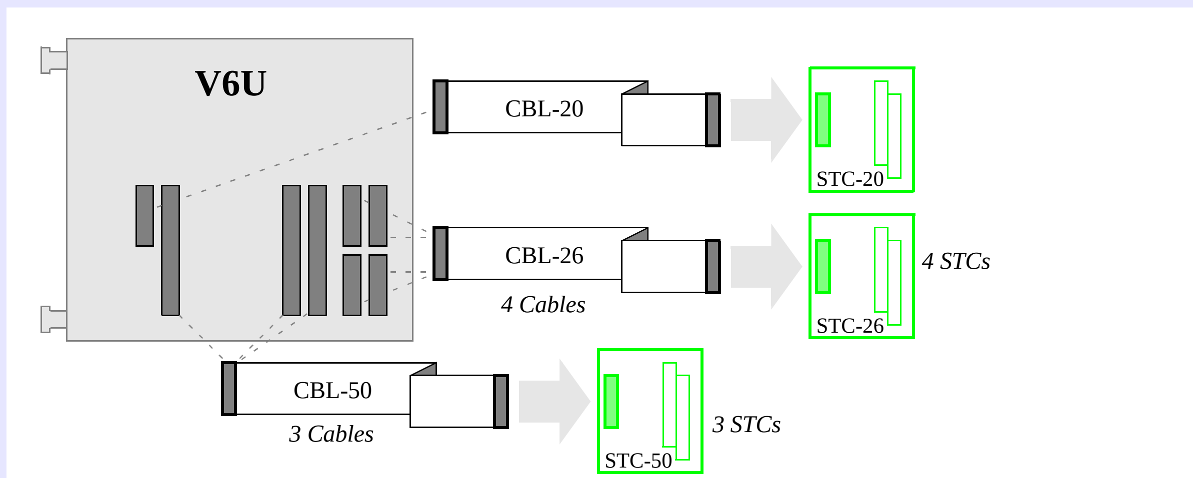
The IRQ Level switch configures the V6U's on-board logic to decode an interrupt acknowledgment from the host processor. The switch settings correspond to a binary representation of the particular IRQ line connected by the IRQ Select switch.

For example, if IRQ3 is selected, then SW4 should represent the decimal value 3. So, SW4-1 and SW4-2 must be ON. SW4-3 and SW4-4 must be OFF.

Table 2-13 IRQ Level Switch SW4

IRQ Level	*4*	*3*	*2*	*1*	
LEVEL 0	off	off	off	off	*Default*
LEVEL 1	off	off	off	ON	
LEVEL 2	off	off	ON	off	
LEVEL 3	off	off	ON	ON	
LEVEL 4	off	ON	off	off	
LEVEL 5	off	ON	off	ON	
LEVEL 6	off	ON	ON	off	
LEVEL 7	off	ON	ON	ON	

Connect Cables/Insert Board



Connect V6U to STCs

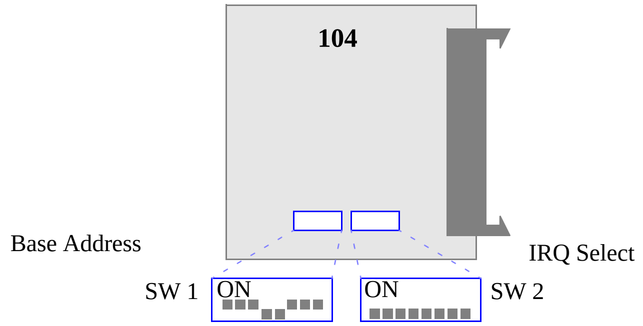
To install the controller:

1. **Turn off the power** to the VME chassis.
2. Select an unused slot.
3. Install all required ribbon cables.
4. Insert the controller and press firmly until the board is seated in the backplane connector.
5. Fasten the mounting screws (for models that have mounting screws).
6. Proceed to Chapter 3 to test your I/O Address.

104

Locate Switches

Figure 2-6 104 Address and IRQ Switch Locations



104 Switches

Set Base I/O Address

Use the SW1 dipswitch on each controller to set the base I/O address.

Table 2-14 Base Address Switch SW1

Address	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	
240	on	on	OFF	on	on	OFF	on	on	
250	on	on	OFF	on	on	OFF	on	OFF	
260	on	on	OFF	on	on	OFF	OFF	on	
270	on	on	OFF	on	on	OFF	OFF	OFF	
300	on	on	OFF	OFF	on	on	on	on	*Default*
310	on	on	OFF	OFF	on	on	on	OFF	
320	on	on	OFF	OFF	on	on	OFF	on	
330	on	on	OFF	OFF	on	on	OFF	OFF	
340	on	on	OFF	OFF	on	OFF	on	on	
350	on	on	OFF	OFF	on	OFF	on	OFF	
360	on	on	OFF	OFF	on	OFF	OFF	on	
370	on	on	OFF	OFF	on	OFF	OFF	OFF	

on = low
OFF = high

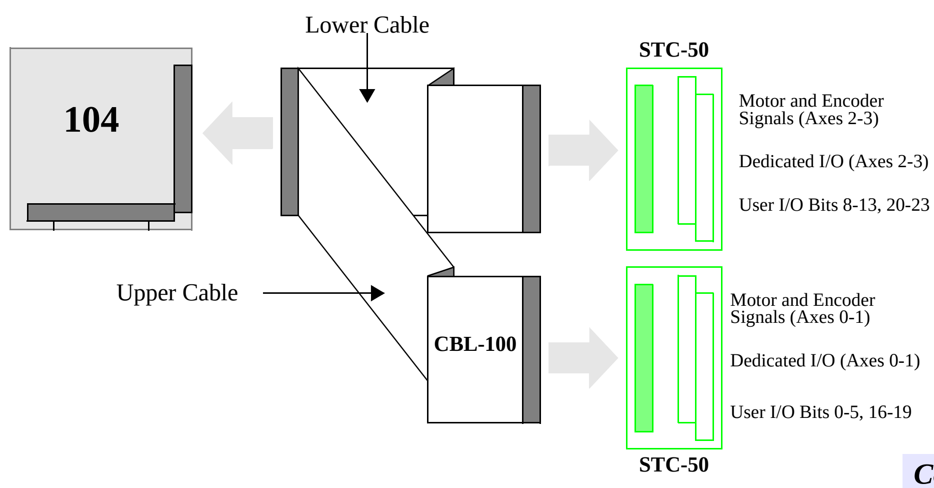
Set the Interrupts

Interrupts may be generated from the DSP Series controller to the host CPU. SW2 connects the controller's interrupt circuitry to one of the host CPU's IRQ lines. To use one of the IRQ lines, you must configure switch SW2. Configure switch SW2 for the interrupt (IRQ2, IRQ3, ...) that you want the 104 to use.

Table 2-15 IRQ Switch SW2

IRQ	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	
None	off	off	off	off	off	off	off	off	*Default*
IRQ2	off	off	off	off	off	off	off	ON	
IRQ3	off	off	off	off	off	off	ON	off	
IRQ4	off	off	off	off	off	ON	off	off	
IRQ5	off	off	off	off	ON	off	off	off	
IRQ10	off	off	off	ON	off	off	off	off	
IRQ11	off	off	ON	off	off	off	off	off	
IRQ12	off	ON	off	off	off	off	off	off	
IRQ15	ON	off	off	off	off	off	off	off	

Connect Cables/Insert Board



Connect 104 to STCs

To install the controller:

1. **Turn off the power** to your computer.
2. Insert the controller and press firmly until the board is seated.
3. Secure the standoffs in place.
4. Connect the CBL-100. The 100-pin high density connector fits into the 104 controller locking header. The two 50-pin connectors fit into the locking headers on the STC-50s. (STC-50 shown above, see *Appendix F, OptoCon Reference* if using the Opto-Con).

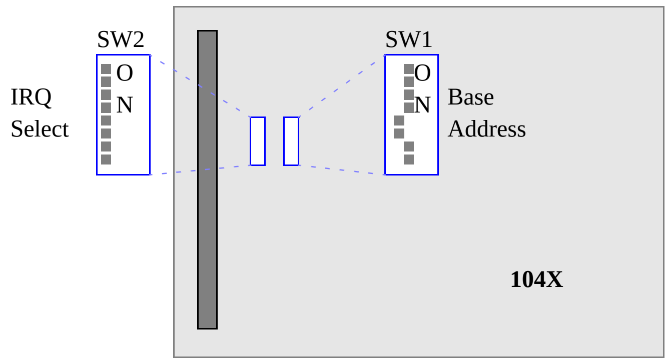
For 104 Users	For an easy way to separate 104 cards, get the **PC/104 Removal Tool**, available from: Enclosure Technologies 256 Airport Industrial Blvd. Ypsilanti, MI 48198 phone: 313-481-2200

5. Proceed to Chapter 3 to test your I/O Address.

104X

Locate Switches

Figure 2-7 104X Address and IRQ Switch Locations



104X Switches

Set Base I/O Address

Use the SW1 dipswitch on each controller to set the base I/O address.

Table 2-16 Base Address Switch SW1

Address	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*
240	on	on	OFF	on	on	OFF	on	on
250	on	on	OFF	on	on	OFF	on	OFF
260	on	on	OFF	on	on	OFF	OFF	on
270	on	on	OFF	on	on	OFF	OFF	OFF
300	on	on	OFF	OFF	on	on	on	on
310	on	on	OFF	OFF	on	on	on	OFF
320	on	on	OFF	OFF	on	on	OFF	on
330	on	on	OFF	OFF	on	on	OFF	OFF
340	on	on	OFF	OFF	on	OFF	on	on
350	on	on	OFF	OFF	on	OFF	on	OFF
360	on	on	OFF	OFF	on	OFF	OFF	on
370	on	on	OFF	OFF	on	OFF	OFF	OFF

Default (300)

on = low
OFF = high

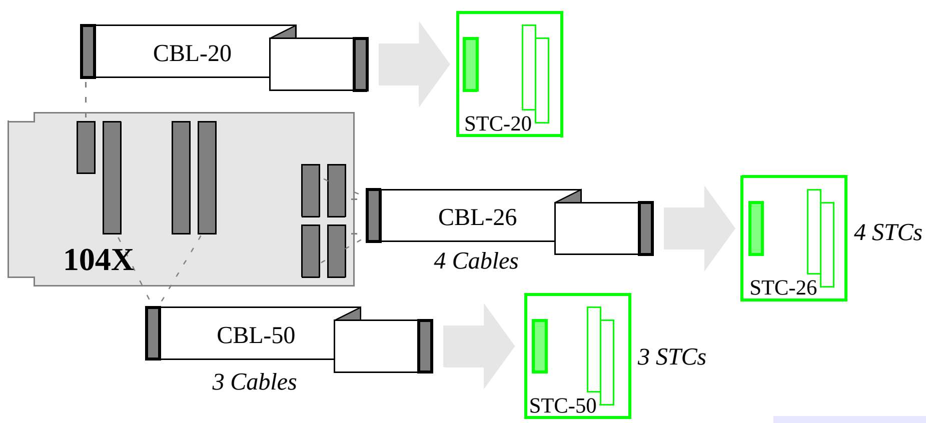
Set the Interrupts

Interrupts may be generated from the DSP Series controller to the host CPU. SW2 connects the controller’s interrupt circuitry to one of the host CPU’s IRQ lines. To use one of the IRQ lines, you must configure switch SW2. Configure switch SW2 for the interrupt (IRQ2, IRQ3, ...) that you want the 104X to use.

Table 2-17 IRQ Switch SW2

IRQ	8	7	6	5	4	3	2	1	
None	off	off	off	off	off	off	off	off	*Default*
IRQ2	off	off	off	off	off	off	off	ON	
IRQ3	off	off	off	off	off	off	ON	off	
IRQ4	off	off	off	off	off	ON	off	off	
IRQ5	off	off	off	off	ON	off	off	off	
IRQ10	off	off	off	ON	off	off	off	off	
IRQ11	off	off	ON	off	off	off	off	off	
IRQ12	off	ON	off	off	off	off	off	off	
IRQ15	ON	off	off	off	off	off	off	off	

Connect Cables/Insert Board



Connect 104X to STCs

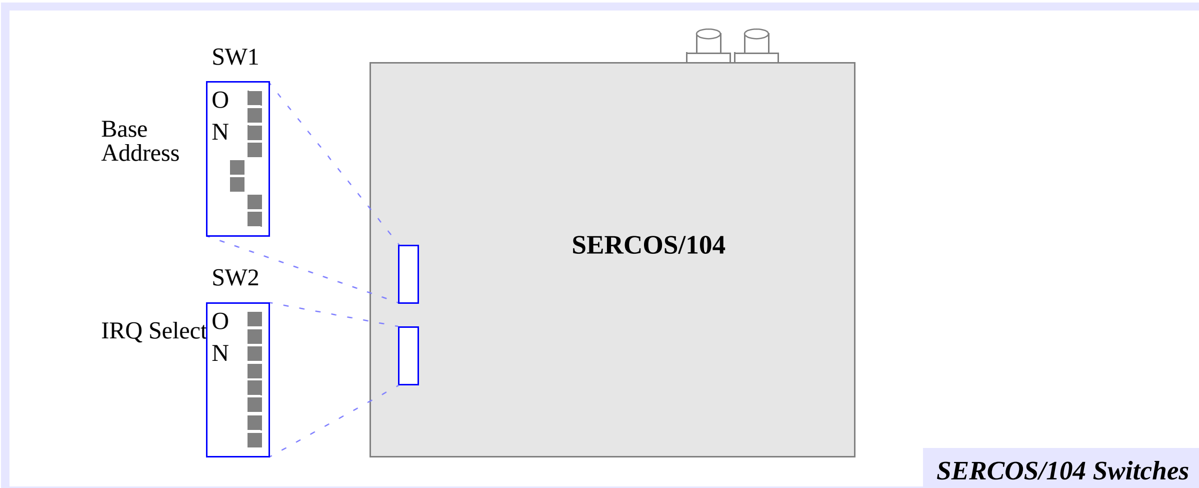
To install the controller:

1. **Turn off the power** to your computer.
2. Insert the controller (8 or 16-bit) and press firmly until the board is seated.
3. Secure the standoffs in place.
4. Install all ribbon cables.
5. Proceed to Chapter 3 to test your I/O Address.

SERCOS/104

Locate Switches

Figure 2-8 SERCOS/104 Address and IRQ switch locations



Set the Base I/O Address

Use the SW1 dipswitch on each controller to set the base I/O address.

Table 2-18 Base Address Switch SW1

Address	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	
240	on	on	OFF	on	on	OFF	on	on	
250	on	on	OFF	on	on	OFF	on	OFF	
260	on	on	OFF	on	on	OFF	OFF	on	
270	on	on	OFF	on	on	OFF	OFF	OFF	
300	on	on	OFF	OFF	on	on	on	on	*Default*
310	on	on	OFF	OFF	on	on	on	OFF	
320	on	on	OFF	OFF	on	on	OFF	on	
330	on	on	OFF	OFF	on	on	OFF	OFF	
340	on	on	OFF	OFF	on	OFF	on	on	
350	on	on	OFF	OFF	on	OFF	on	OFF	
360	on	on	OFF	OFF	on	OFF	OFF	on	
370	on	on	OFF	OFF	on	OFF	OFF	OFF	

on = low
OFF = high

Set the Interrupts

Interrupts may be generated from the DSP Series controller to the host CPU. SW2 connects the controller's interrupt circuitry to one of the host CPU's IRQ lines. To use one of the IRQ lines, you must configure switch SW2. Configure switch SW2 for the interrupt (IRQ2, IRQ3, ...) that you want the SERCOS/104 to use.

Table 2-19 IRQ Switch SW2

IRQ	***8***	***7***	***6***	***5***	***4***	***3***	***2***	***1***	
None	off	off	off	off	off	off	off	off	*Default*
IRQ2	off	off	off	off	off	off	off	ON	
IRQ3	off	off	off	off	off	off	ON	off	
IRQ4	off	off	off	off	off	ON	off	off	
IRQ5	off	off	off	off	ON	off	off	off	
IRQ10	off	off	off	ON	off	off	off	off	
IRQ11	off	off	ON	off	off	off	off	off	
IRQ12	off	ON	off	off	off	off	off	off	
IRQ15	ON	off	off	off	off	off	off	off	

Connect Cables/Insert Board

To install the controller:

1. **Turn off the power** to the computer.
2. Select an unused slot (8 or 16-bit).
3. Insert the controller and press firmly until the board is seated in the card-edge connector.
4. Connect the fiber optic cables in a ring between the SERCOS/104 and the drives. The dark gray connectors are receivers (“Rx”) and the light gray connectors are transmitters (“Tx”). Connect the controller’s light gray connector to the first drive’s dark gray (Rx) connector, the connect the first drive’s light gray (Tx) connector to the second drive’s dark gray (Rx) connector, etc.

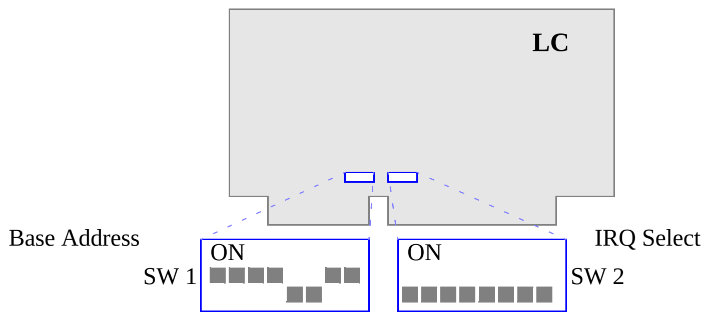
   The light-emitting module on the controller can be turned on and off for testing with the functions **turn_on_sercos_led(…)** and **turn_off_sercos_led(…).** (See the *DSP Series C Programming Reference* for more information.)

Once the SERCOS/104 has been initialized, you can exercise and tune the system using Motion Console.

LC

Locate Switches

Figure 2-9 LC Address and IRQ Switch Locations



LC Switches

Set Base I/O Address

Use the SW1 dipswitch on each controller to set the base I/O address.

Table 2-20 Base Address Switch SW1

Address	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	
240	on	on	OFF	on	on	OFF	on	on	
250	on	on	OFF	on	on	OFF	on	OFF	
260	on	on	OFF	on	on	OFF	OFF	on	
270	on	on	OFF	on	on	OFF	OFF	OFF	
300	on	on	OFF	OFF	on	on	on	on	*Default*
310	on	on	OFF	OFF	on	on	on	OFF	
320	on	on	OFF	OFF	on	on	OFF	on	
330	on	on	OFF	OFF	on	on	OFF	OFF	
340	on	on	OFF	OFF	on	OFF	on	on	
350	on	on	OFF	OFF	on	OFF	on	OFF	
360	on	on	OFF	OFF	on	OFF	OFF	on	
370	on	on	OFF	OFF	on	OFF	OFF	OFF	

on = low
OFF = high

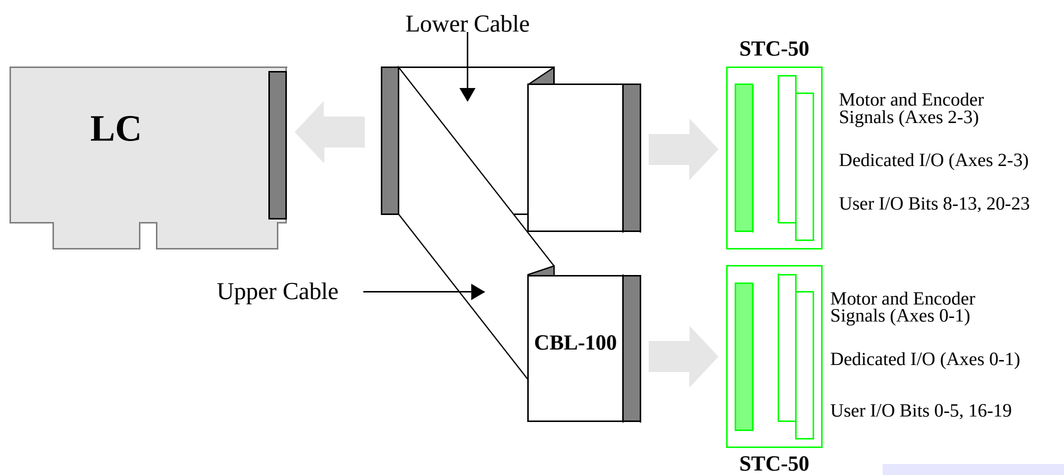
Set the Interrupts

Interrupts may be generated from the DSP Series controller to the host CPU. SW2 connects the controller's interrupt circuitry to one of the host CPU's IRQ lines. To use one of the IRQ lines, you must configure switch SW2. Configure switch SW2 for the interrupt (IRQ2, IRQ3, ...) that you want the LC to use.

Table 2-21 IRQ Switch SW2

IRQ	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	
None	off	off	off	off	off	off	off	off	*Default*
IRQ2	off	off	off	off	off	off	off	ON	
IRQ3	off	off	off	off	off	off	ON	off	
IRQ4	off	off	off	off	off	ON	off	off	
IRQ5	off	off	off	off	ON	off	off	off	
IRQ10	off	off	off	ON	off	off	off	off	
IRQ11	off	off	ON	off	off	off	off	off	
IRQ12	off	ON	off	off	off	off	off	off	
IRQ13	ON	off	off	off	off	off	off	off	

Connect Cables/Insert Board



Connect LC to STCs

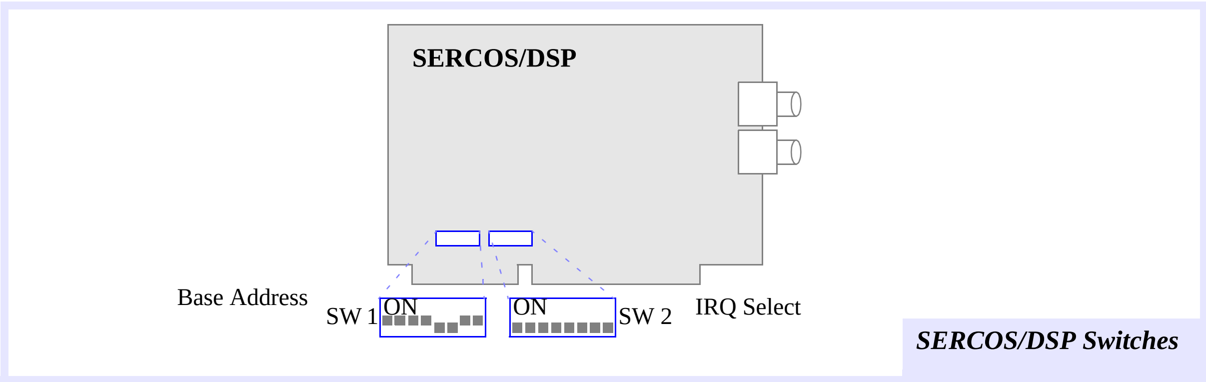
To install the controller:

1. **Turn off the power to the computer** and remove the cover.
2. Select an unused expansion slot (16-bit) and remove its blank metal bracket from the computer.
3. Orient the controller inside the computer so that it lines up with the card-edge connector.
4. Press down on the metal bracket tab and the top of the board until the board is firmly seated.
5. Secure the bracket in place with the screw.
6. Connect the CBL-100. The 100-pin high density connector fits into the LC controller locking header. The two 50-pin connectors fit into the locking headers on the STC-50s. (STC-50 shown above, see *Appendix F, OptoCon Reference* if using the OptoCon).
7. Proceed to Chapter 3 to test the I/O Address.

SERCOS/DSP

Locate Switches

Figure 2-10 SERCOS/DSP Address and IRQ Switch Locations



Set Base I/O Address

Use the SW1 dipswitch on each controller to set the base I/O address.

Table 2-22 Base Address Switch SW1

Address	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	
240	on	on	OFF	on	on	OFF	on	on	
250	on	on	OFF	on	on	OFF	on	OFF	
260	on	on	OFF	on	on	OFF	OFF	on	
270	on	on	OFF	on	on	OFF	OFF	OFF	
300	on	on	OFF	OFF	on	on	on	on	*Default*
310	on	on	OFF	OFF	on	on	on	OFF	
320	on	on	OFF	OFF	on	on	OFF	on	
330	on	on	OFF	OFF	on	on	OFF	OFF	
340	on	on	OFF	OFF	on	OFF	on	on	
350	on	on	OFF	OFF	on	OFF	on	OFF	
360	on	on	OFF	OFF	on	OFF	OFF	on	
370	on	on	OFF	OFF	on	OFF	OFF	OFF	

on = low
OFF = high

Set the Interrupts

Interrupts may be generated from the DSP Series controller to the host CPU. SW2 connects the controller's interrupt circuitry to one of the host CPU's IRQ lines. To use one of the IRQ lines, you must configure switch SW2. Configure switch SW2 for the interrupt (IRQ2, IRQ3, ...) that you want the SERCOS/DSP to use.

Motion Engineering, Inc.

Table 2-23 IRQ Switch SW2

IRQ	***8***	***7***	***6***	***5***	***4***	***3***	***2***	***1***	
None	off	off	off	off	off	off	off	off	*Default*
IRQ2	off	off	off	off	off	off	off	ON	
IRQ3	off	off	off	off	off	off	ON	off	
IRQ4	off	off	off	off	off	ON	off	off	
IRQ5	off	off	off	off	ON	off	off	off	
IRQ10	off	off	off	ON	off	off	off	off	
IRQ11	off	off	ON	off	off	off	off	off	
IRQ12	off	ON	off	off	off	off	off	off	
IRQ15	ON	off	off	off	off	off	off	off	

Connect Cables/Insert Board

To install the controller:

1. **Turn off the power to the computer** and remove the cover.
2. Select an unused expansion slot (16-bit) and remove its blank metal bracket from the computer.
3. Orient the controller inside the computer so that it lines up with the card-edge connector.
4. Press down on the metal bracket tab and the top of the controller until the board is firmly seated.
5. Secure the bracket in place with the screw.
6. Connect the fiber optic cables in a ring between the SERCOS/DSP and the drives. The dark gray connectors are receivers (“Rx”) and the light gray connectors are transmitters (“Tx”). Connect the controller’s light gray connector to the first drive’s dark gray (Rx) connector, the connect the first drive’s light gray (Tx) connector to the second drive’s dark gray (Rx) connector, etc.

   The light emitting module on the controller can be turned on and off for testing with the functions **turn_on_sercos_led(…)** and **turn_off_sercos_led(…).** (See the *DSP Series C Programming Manual* for more information).

7. When all drives are connected, turn on the power to the drives. Each drive begins an initialization sequence. Most drives have an LCD or LED display to indicate when the initialization is complete. Consult the specific drive documentation and *DSP Series C Programming Refernce* for more information about SERCOS initialization procedures.

Once the SERCOS/DSP has been initialized, you can exercise and tune the system using Motion Console.

CHAPTER 3 TEST CONTROLLER'S I/O ADDRESS

Now before wiring the STCs to the amplifiers, encoders or motors, test the I/O address of the DSP Series controller.

If your Operating System is	*Then use this application to test the I/O location*	
Windows 95/98 Windows NT Windows (with 32S extensions)	Motion Console	3-2
DOS Windows 3.11	SETUP or CONFIG	3-3 3-4

If your systems is running Windows 95/98, Windows NT, and Windows (with 32S extensions), you can use *Motion Console* to test the I/O location.

If your systems is running DOS or Windows 3.11, you must use the SETUP program (or the CONFIG program) to test the I/O location.

After testing your controller's I/O address, proceed to Chapter 4, to connect the STCs to the amplifiers, motors and encoders.

Warning!	Only use Motion Console version 2.00.0006 or later with the PCI/DSP (some required features are not included in prior versions).

Using Motion Console

1. To install MEI's Motion Console application, follow the instructions in the Release Note included with your software distribution.

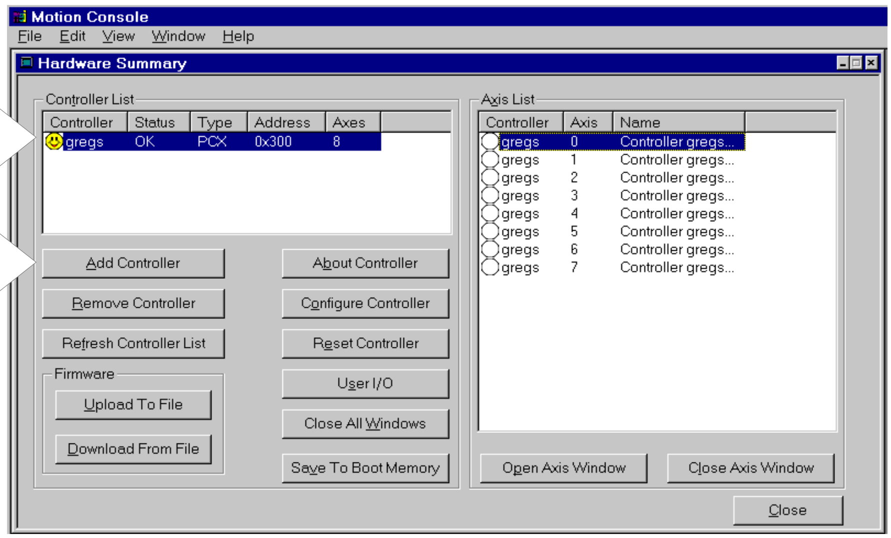
2. Locate the Motion Console application, which should be located in the Motion Engineering program group (\MEI). Start Motion Console by clicking on its icon.

3. In Motion Console's main menu, select *Summary*. You should now see the Hardware Summary window.

4. Click *Add Controller* in the *Hardware Summary* window.

Controller's status

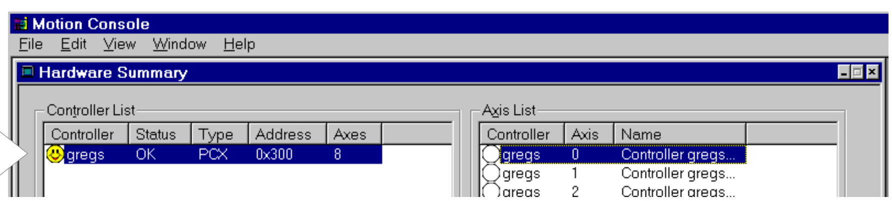
To change the controller's I/O address



5. In the *Add Controller* dialog box, enter the controller's name and desired address (for PCI/DSP click on the *PCI Controllers* tab and select controller). Next click *OK*.

6. The new controller should now appear in the *Hardware Summary*/*Controller List* with a status of "OK." If Motion Console cannot find the controller at the specified address, Motion Console will list the controller's status as "Bad."

Controller should appear here



If the controller's name and address appear as desired, proceed to Chapter 4 and continue with your installation, by connecting the STCs to amplifiers, encoders and motors.

If the controller's name and address appear as desired or if Motion Console lists the controller's status as "Bad" in the *Controller List*, make sure that the DIP switches on the controller are correctly set for the same address.

Using SETUP.EXE

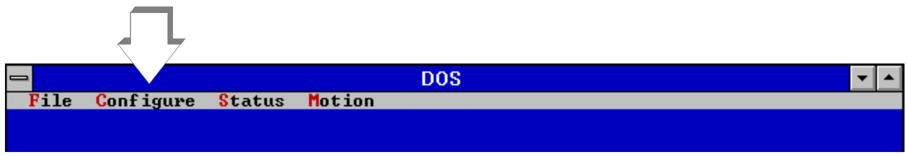
The "Setup" CD-ROM contains the SETUP program, the firmware (.ABS files) and the CONFIG program.

1. On your hard drive (C: or whatever), create the directory C:\MEI\SETUP and copy the files from the "Setup" CD-ROM to that directory.

2. Next run the SETUP program by typing SETUP at the DOS prompt. You should next see the *About SETUP* window, which shows the date and version of the SETUP program.

   Note that when SETUP initializes the controller, SETUP does not change *any* of the current configurations or conditions on the DSP Series controller.

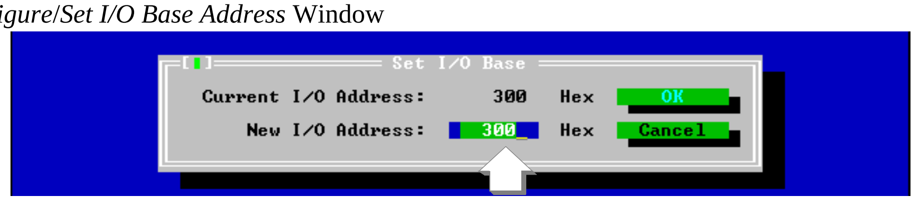
3. Select the *Configure* menu, using either the mouse or by pressing the **ALT** and **C** keys simultaneously. On the *Configure* menu, select *I/O Base Address*.



4. In the *I/O Base Address* window, enter the desired base address for the controller, then select *OK*. Reopen the *I/O Base Address* window, and verify that the *Current I/O Address* is now the address that you just entered.

   If the address is correctly set, then proceed to Chapter 4 and continue with your installation, by connecting the STCs to amplifiers, encoders and motors.

Configure/*Set I/O Base Address* Window



Tip! *DSP Not Found*	If SETUP displays a message that the DSP controller cannot be found at the specified address, be sure that the DIP switches on the controller are set for the same address entered on the CONFIGURE/SET I/O BASE window. If SETUP still displays a message that the *DSP is Not Found*, press the **F9** key to re-execute the SETUP program. If the SETUP program still cannot find the DSP, run the CONFIG program.

Using CONFIG.EXE

The CONFIG program downloads firmware to the controller, configures the DAC offsets, and performs some basic tests of the axes. Normally the CONFIG program is not needed, since the controller is configured at the factory.

WARNING!

Before running CONFIG, disconnect all of the cables from the DSP Series controller and **turn off the power** to any external devices (amplifiers, etc.).

1. To run CONFIG, switch to the directory where all the .ABS files and CONFIG.EXE are stored (generally C:\MEI\SETUP\). Then execute **`config`**.

   CONFIG will download 8AXIS.ABS or 8AXISSER.ABS (for SERCOS controllers).

2. Now execute `config -b` *base_address*

   where *base_address* is the desired I/O address for the DSP controller.

3. If CONFIG doesn't display any error messages, then the I/O address was successfully set.

   Now proceed to Chapter 4 and continue with your installation, by connecting the STCs to amplifiers, encoders and motors.

Other CONFIG Functions

The following CONFIG tests can verify proper communication between the controller and the host CPU, verify on-board memory, configure the DAC offsets, and determine the number of hardware axes. Note that configured DAC offsets **are saved** to the controller's firmware, and are **not saved** to the firmware files on diskette. If there are any problems, the CONFIG program will display error messages.

Table 3-1 CONFIG's Command Line Switches

Configure controller with a particular firmware file	`-f [`*`filename`*`]`
Download firmware file only	`-d [`*`filename`*`]`
Upload firmware file only	`-u [`*`filename`*`]`
Set base address	`-b [`*`base`*`]`
Configure 'n' number of axes	`-a [`*`axes`*`]`
Verbose, all messages displayed	`-v`
No warning message	`-w`

Examples

To configure a controller located at an address other than the default (300 Hex), use the *-b* command line switch. For example, to configure a controller located at address 0x280, execute `CONFIG -B 0x280`.

To download a particular firmware file (.ABS), execute `CONFIG -F MYFIRM.ABS`.
The CONFIG program will download *MYFIRM.ABS* and configure the DAC offsets appropriately. This method is useful for configuring multiple controller cards.

To download firmware without configuring the DAC offsets, execute `CONFIG -D MYFIRM.ABS`.

To upload firmware to a diskette file, execute `CONFIG -U MYFIRM.ABS`.

CHAPTER 4 *CONNECT STCS TO AMPS/MOTOR/ENCODER*

For more information about motor connectors, pinouts on the DSP controllers and signal specifications, please refer to Appendix E, *Connectors & Specifications*.

PCX, STD, 104X, CPCI & V6U

Connections to Servo Motors

DSP Series controllers can control brush servo motors, brushless servo motors, or linear brush/brushless motors. Basic connections require an analog output signal (from the controller to the amplifier) and an encoder input (from the motor to the controller).

Most amplifiers support either Velocity mode (voltage control) Torque mode (current control) or both. The DSP controller can be used with either servo motor/amplifier package.

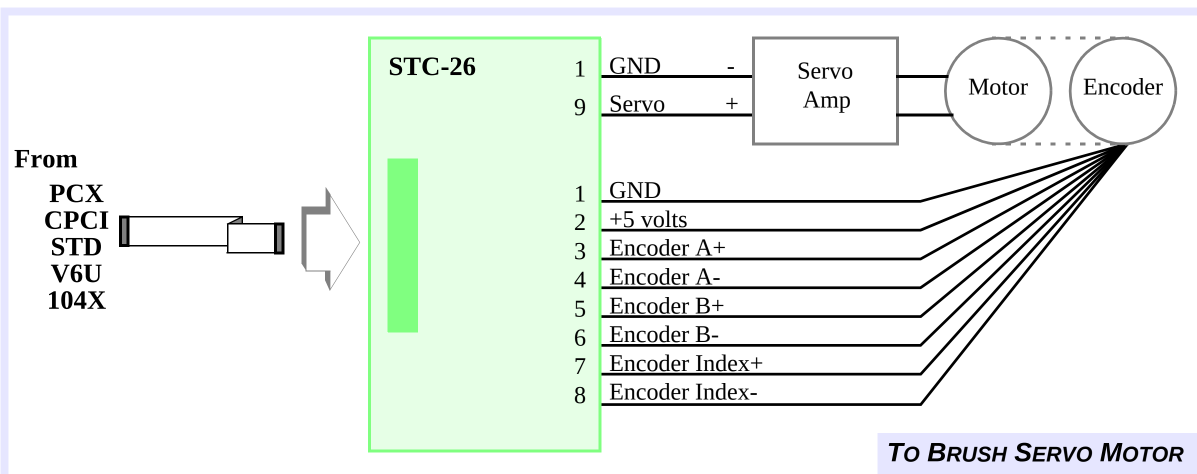
DSP Series controllers accept TTL-level (0V to +5V, 40mA max) encoder input from either differential or single-ended encoders. Differential encoders are preferred due to their excellent noise immunity. The connections for a single-ended encoder are identical to a differential encoder except that no connections should be made to channel A- and channel B-. (The A- and B- lines are pulled up internally to +2.5V). **Single-ended encoder connections are different for the V6U, see page 7 in this chapter for V6U connections.**

The controller reads the index pulse (either single-ended or differential ended). Typically, there is one index pulse per revolution of the encoder (rotary type), which can be used for homing. Encoder signals are read in quadrature. Every line on the encoder will produce a rising edge and a falling edge on channels A+ and B+ which is interpreted by the DSP controller as 4 encoder counts.

Brush Servo Motors

The minimum required connections to brush-type servo are: Analog signal (+/- 10V), +5V, Signal Ground, Encoder Channel A+, Encoder Channel B+. Typical connections for a brush servo motor with a differential encoder are:

Figure 4-1 Typical Brush Servo Motor Connections



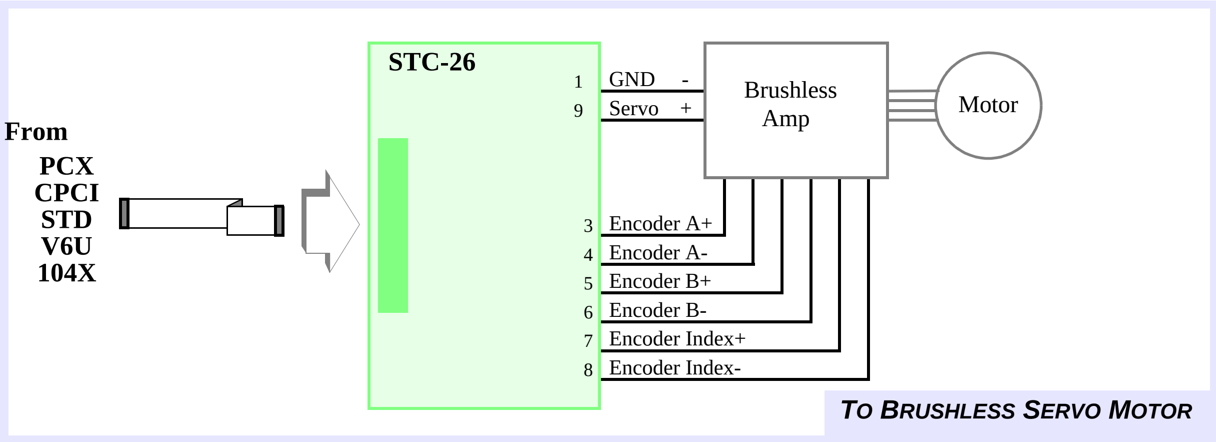
For more information about amp enable and amp fault connections, see the *Dedicated & User I/O* section in Chapter 5.

Note	**Any unused lines should be left unconnected.**

Brushless Servo Motors

Typical connections for a brushless servo motor with a differential encoder are:

Figure 4-2 Typical Brushless Servo Motor Connections



For more information about amp enable and amp fault connections, see the *Dedicated & User I/O* section in Chapter 5.

Note	**Any unused lines should be left unconnected.**

Step-and-Direction Controlled Servo Motors

Some brushless servos are controlled by step-and-direction pulses. With this scheme, the position information is communicated by step pulses, and the PID loop is handled internally by the drive itself.

To avoid possible instability caused by conflict between the drive PID loop and the controller board PID loop, **operate step-and-direction servos as open-loop step motors**. The controller will send step pulses and a direction pulse to the drive, which will handle the PID internally.

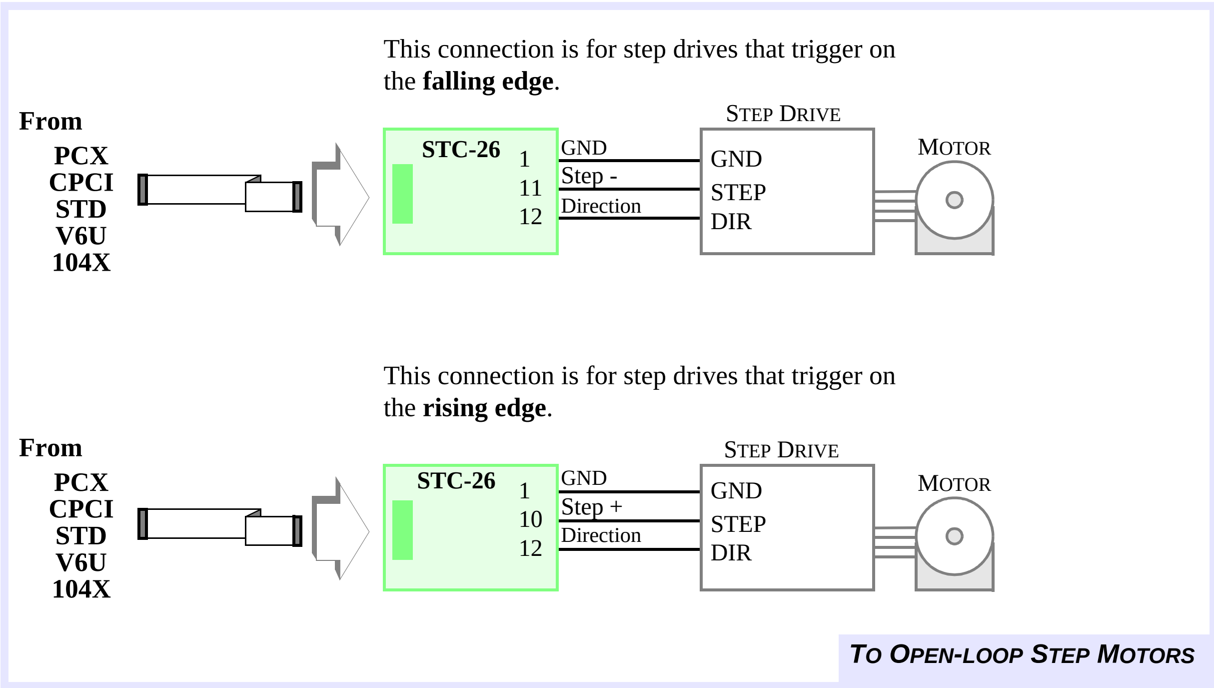
Warning!	If the controller is configured for open loop step control, make sure that the tuning parameters conform to those listed in *Open-Loop Stepper Systems* (Chapter 6).

Connections to Step Motors

Open-Loop Step Motors

The DSP controllers can control step motors in both open-loop (no encoder) and closed-loop configurations. In the open-loop configuration the step pulse output (connected to the drive) is fed back into the line receivers and used to keep track of the "actual position." With open-loop step configuration selected, the DSP closes the loop internally on a pair of axes. DSP controllers are compatible with full, half and micro stepping drives.

Figure 4-3 Typical Open-Loop Step Motor Connections (PCX/CPCI/STD/V6U/104X)



Closed-Loop Step Motors

DSP Series controllers can control step motors with encoder feedback. Closed-loop steps are controlled by a PID algorithm running on the DSP in real time. The controllers accept TTL-level (0V to 5V, 40mA max) encoder input from either differential or single-ended encoders. Differential encoders are preferred due to their excellent noise immunity. The connections for a single-ended encoder is identical to a differential encoder except that there are no connections made to channel A- and channel B-. (The A- and B- lines are pulled up internally to 2.5V).

Encoder signals are read in quadrature. Every line on the encoder will produce a rising edge and a falling edge on channels A+ and B+ which is interpreted by the DSP controller as 4 encoder counts.

Connecting closed-loop step motors to the controller is similar to servo motors, except that the step and direction lines are connected instead of the analog signal. The minimum connections are:

Step+ (or Step-)
Direction+ (or Direction-)
Signal Ground
Encoder A+ and B+ lines
+5V

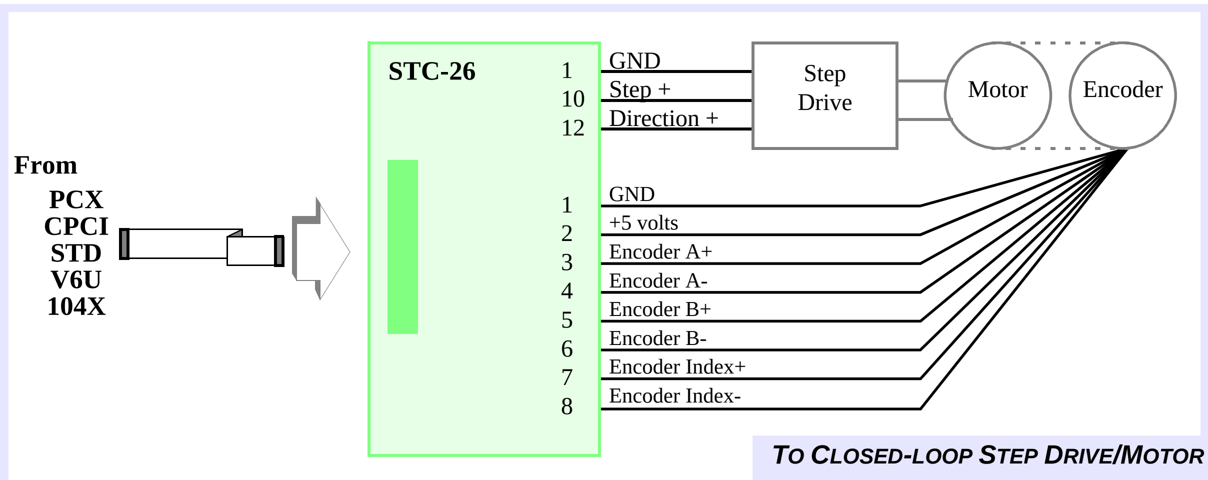
Note that when only *Step+* or *Step-* is used, it may be necessary *to jumper unused terminals on the step drive*. Before connecting *Step+* or *Step-*, consult your step drive's manual

In general, use *Step+* for drives with active high logic, and use *Step-* for drives with active low logic. Both *Step+* and *Step-* lines can be connected to drives with differential inputs. If in doubt, fax the drive pinouts to Motion Engineering along with any questions.

Warning!	For the best performance, ensure that the ratio is between the *encoder resolution* (counts per revolution) and the *step resolution* (steps per microsteps per revolution) is 1:4. Lower ratios (1:1, 1:2) will be difficult to tune and will have poor static stability. Higher ratios (1:6, 1:8) will have poor constant velocity stability.

Typical connections for a step motor with a differential encoder are:

Figure 4-4 Typical Closed-loop Step Motor Connections



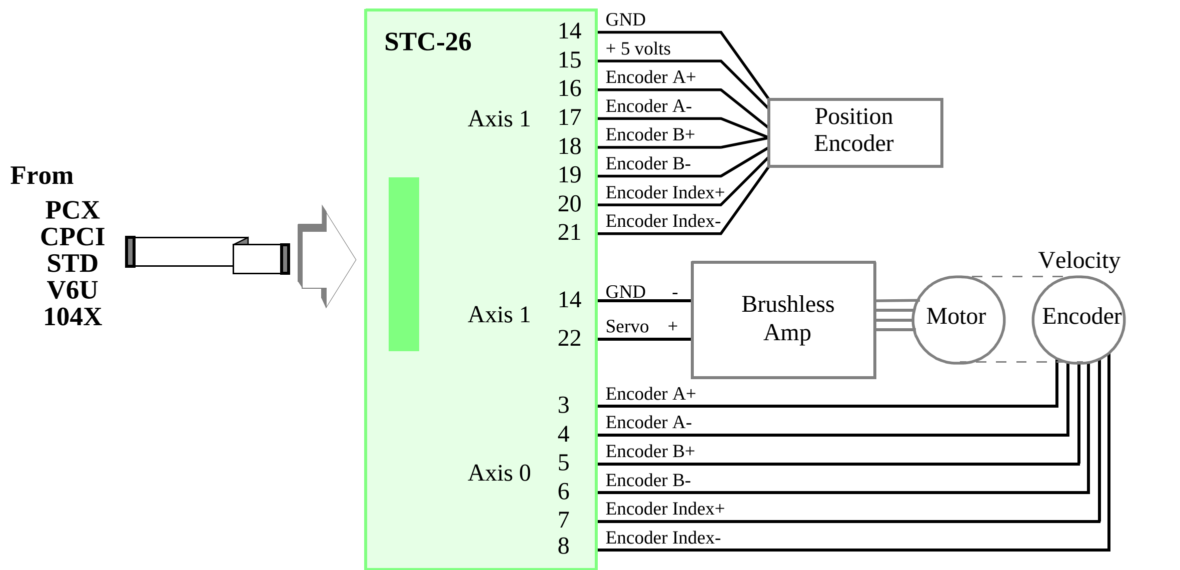
Note!	For drives that trigger on the rising edge of the pulse input, use *Step+*. For drives that trigger on the falling edge of the pulse input, use *Step-*.

Connections for Dual-Loop Control

DSP Series controllers can be configured for dual-loop control. In dual-loop control, the *velocity information* for the PID derivative term (Kd) is derived from a rotary *encoder on the motor* shaft, and the *position information* for the PID proportional and integral terms is derived from an *encoder on the load* itself.

The axis that will be used for the rotary encoder is configurable through software and can be any axis that is not controlling a motor. For example, if axis 0 is configured for velocity feedback and axis 1 is configured for positional feedback, your system would be connected as shown in the next figure.

Figure 4-5 Typical Dual-loop Encoder Connections with Differential Encoders



DUAL-LOOP ENCODER CONNECTIONS

V6U

Encoder Interface

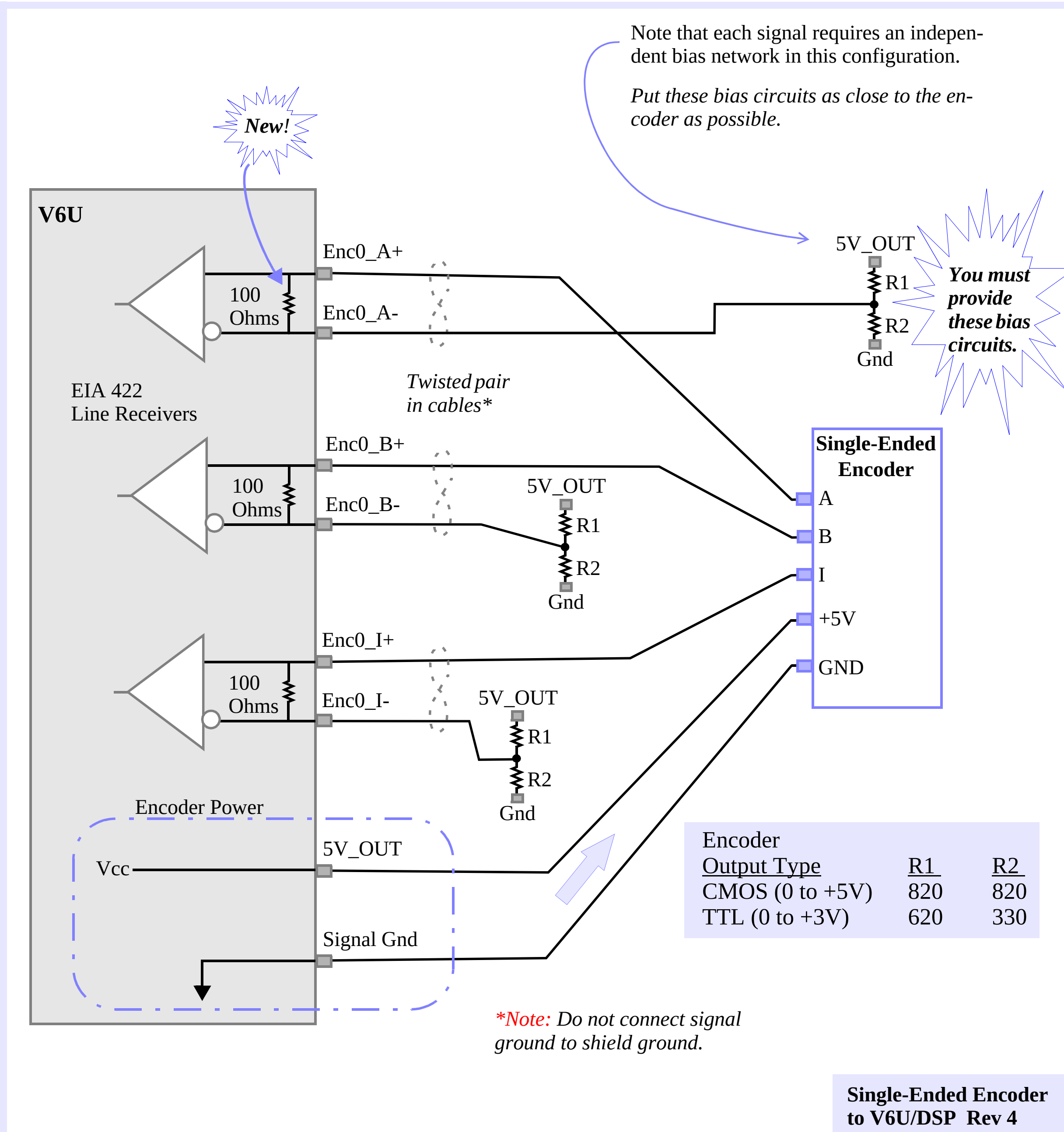
Warning!	**The encoder interface circuits have changed from Revision 2 to Revision 4!** If you use a Rev 4 V6U controller in a “Rev 2” system containing single-ended encoders, the motors may run away and cause harm or injury to equipment and people. **If you are using single-ended encoders with the V6U, you must change the circuitry to work safely with Revision 4 V6U controllers.**

When we added the *Encoder Integrity Checking* feature, we removed the R1/R2 bias circuits and added 100 ohm resistors across the 422 receiver inputs.

If you are using single-ended encoders, you now must add your own bias circuits to your system. The bias circuits are no longer provided on the V6U controller. Differential encoders are connected in the same manner as in previous revisions of the V6U. Connection diagrams for Rev 4 are included here to highlight the wiring changes.

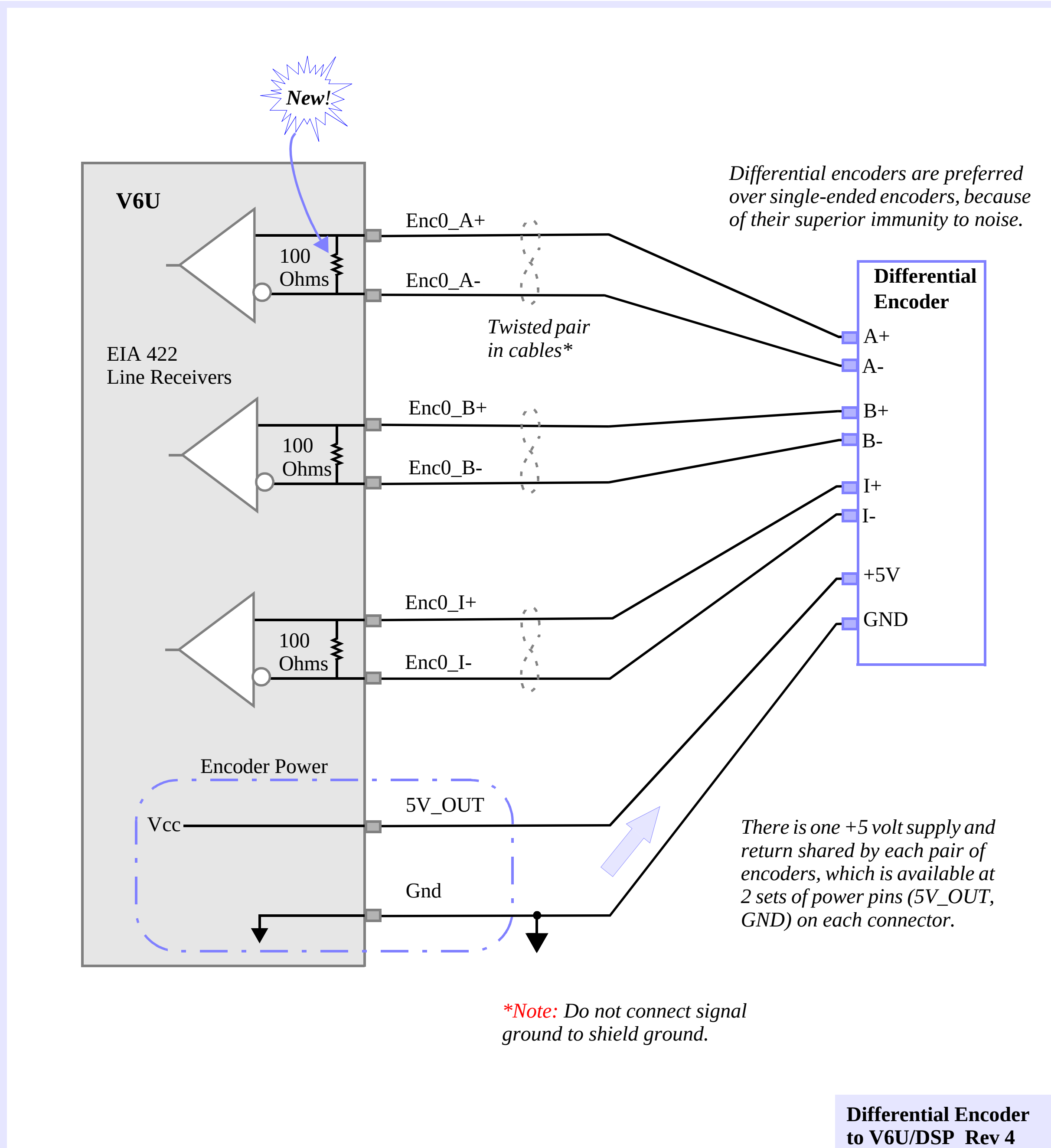
Note that twisted-pair shielded cabling provides the best immunity in electrically noisy environments. For more about *Encoder Integrity Checking*, please consult the DSP Series *C Programming Reference.*

Figure 4-6 Example of Single-Ended Encoder Connection to V6U Rev 4



Encoder Output Type	R1	R2
CMOS (0 to +5V)	820	820
TTL (0 to +3V)	620	330

Figure 4-7 Example of Differential Encoder Connection to V6U Rev 4



Encoder Integrity Checking

V6U Revision 4 now includes broken wire detection and illegal state detection (using digital filtering on encoder input lines). Linear Tech LTC1519 EIA-422 line receivers (with open and short circuit guaranteed states) are used in a flip-flop structure to provide information to existing Encoder Integrity Checking (EIC) logic on the V6U.

Broken Wire & Illegal State Detection

The encoder inputs (channel A+, A-, B+, B-) are monitored by the FPGA (an on-board logic component). The encoder inputs are sampled at 10mHz. A digital filter has been added to each of the encoder inputs to the position counters in the FPGA. This digital filter requires that an encoder input (channel A+, A-, B+, B-) be stable *for 4 clock cycles* (400 nanoseconds) before a transition is recognized, i.e., encoder input states lasting less than 4 clock cycles are considered illegal and filtered out.

A broken wire condition occurs when either (A+ and A- channels) or (B+ or B- channels) are in the same logic state for 3 consecutive sample periods (300 nsec). When a broken encoder wire is detected, the appropriate bit (one per axis) in the broken wire status register is latched.

Use the routine *set_feedback_check(int16 **axis**, int16 ***state**)* to configure broken wire and illegal state detection. To enable feedback checking, set ***state*** = TRUE; to disable feedback checking, set ***state*** = FALSE. After feedback checking is enabled, use the routine *get_feedback_check(int16 **axis**, int16 ***state**)* to read the current feedback checking configuration for an axis. When feedback checking is enabled, the V6U will examine the broken wire and illegal state registers at every DSP sample. If the DSP detects an encoder failure, an *Abort Event* will be generated on the appropriate axis.

Use *axis_source(...)* to determine the cause of the *Exception Event.*

To clear a broken encoder wire or illegal state condition, call *controller_run(...).* This function will clear the broken wire, illegal state registers, and the *Abort Event.*

LC, 104

Connections to Servo Motors

DSP Series controllers can control brush servo motors, brushless servo motors, or linear brushless motors. Basic connections require an analog output signal (from the controller to the amplifier) and an encoder input (from the motor to the controller).

Most amplifiers support either Velocity mode (voltage control), Torque mode (current control) or both. The DSP controller can be used with either Velocity or Torque controlled servo motor/ amplifier packages.

DSP Series controllers accept TTL-level (0V to $^{+}$5V, 40mA max) encoder input from either differential or single-ended encoders. Differential encoders are preferred due to their excellent noise immunity. When used with differential encoders, the differential line receiver on the controller reads the difference between A+ and A- and between B+ and B-. By reading the difference between the square wave inputs any significant noise is canceled out. The connections for a single-ended encoder are identical to a differential encoder except that no connections should be made to channel A- and channel B-. (The A- and B- lines are pulled up internally to 2.5V).

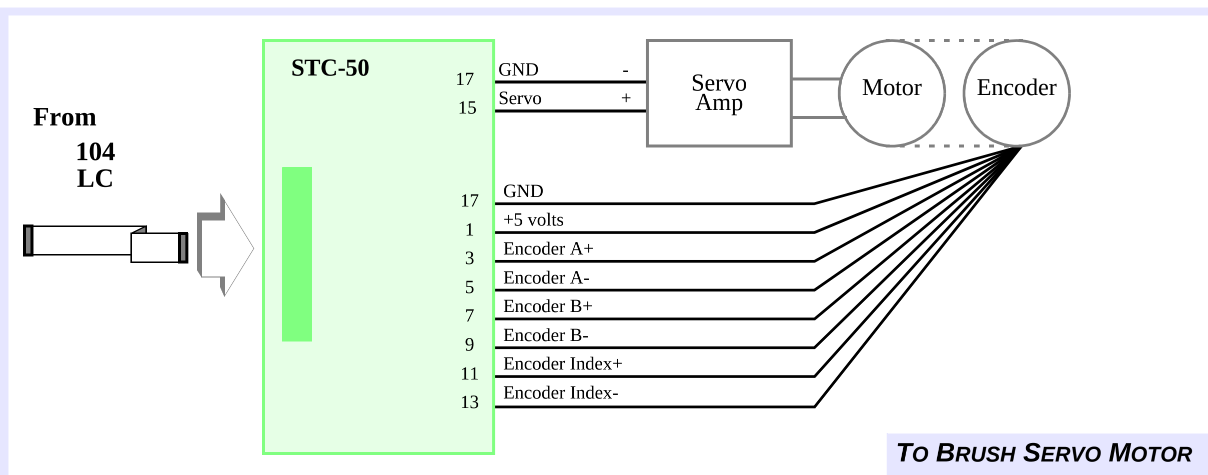
The controller reads the index pulse (either single-ended or differential ended). Typically, there is one index pulse per revolution of the encoder (rotary type), which can be used for homing.

Encoder signals are read in quadrature. Every line on the encoder will produce a rising edge and a falling edge on channels A+ and B+ which is interpreted by the DSP controller as 4 encoder counts.

Brush Servo Motors

The minimum required connections to a brush-type servo are: Analog signal (+/- 10V), +5V, Signal Ground, Encoder Channel A +, Encoder Channel B +. Typical connections for a brush servo motor with a differential encoder are:

Figure 4-8 Typical Brush Servo Motor Connections



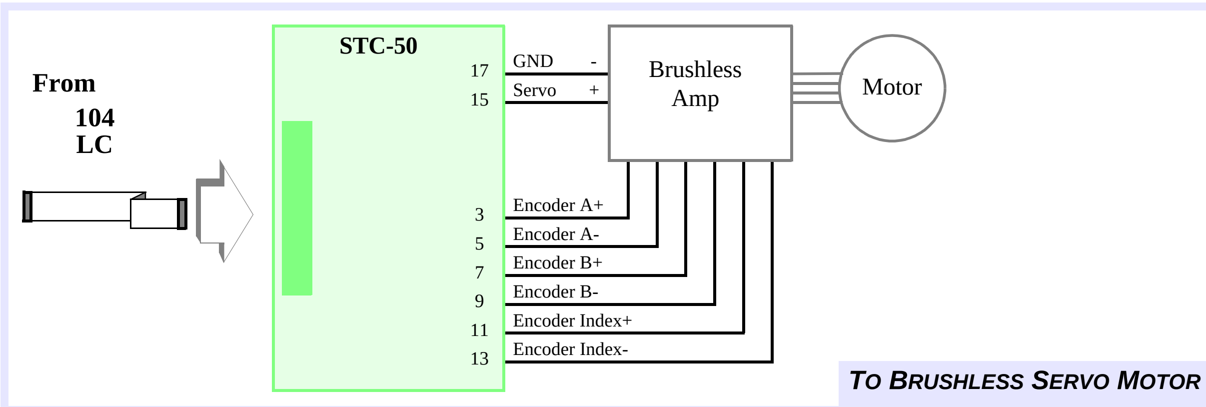
For more information about amp enable and amp fault connections, see the *Dedicated & User I/O* section in Chapter 5.

Note	**Any unused lines should be left unconnected.**

Brushless Servo Motors

Typical connections for a brushless servo motor with a differential encoder are:

Figure 4-9 Typical Brushless Servo Connections



For more information about amp enable and amp fault connections, see the *Dedicated & User I/O* section in Chapter 5.

Note	**Any unused lines should be left unconnected.**

Step-and-Direction Controlled Servo Motors

Some brushless servos are controlled by step-and-direction pulses. With this scheme, the position information is communicated by step pulses, and the PID loop is handled internally by the drive itself.

Step-and-direction servo systems can be operated in open-loop or closed-loop controller configurations. When configured for open-loop steppers, the controller sends step and direction position information to the drive. The drive closes the torque, velocity, and position loops internally. When configured for closed-loop steppers, the controller sends step and direction position information to the drive and receives action position information from the encoder. The drive closes the torque and velocity loops; the controller closes the position loop.

Generally, the best performance occurs when the controller is configured for open-loop steppers.

Note	If the controller is configured for *open loop step control*, make sure that the tuning parameters conform to the parameters listed in *Open-Loop Stepper Systems* (on page 6-7 in Chapter 6).

Connections to Step Motors

Open-Loop Step Motors

The controllers can control step motors in both open-loop (no encoder) and closed-loop configurations. In the open-loop configuration, the step pulse output (connected to the drive) is fed back into the line receivers and used to keep track of the "actual position." With open-loop step configuration selected the DSP closes the loop internally on a pair of axes. DSP controllers are compatible with full/half and micro stepping drives.

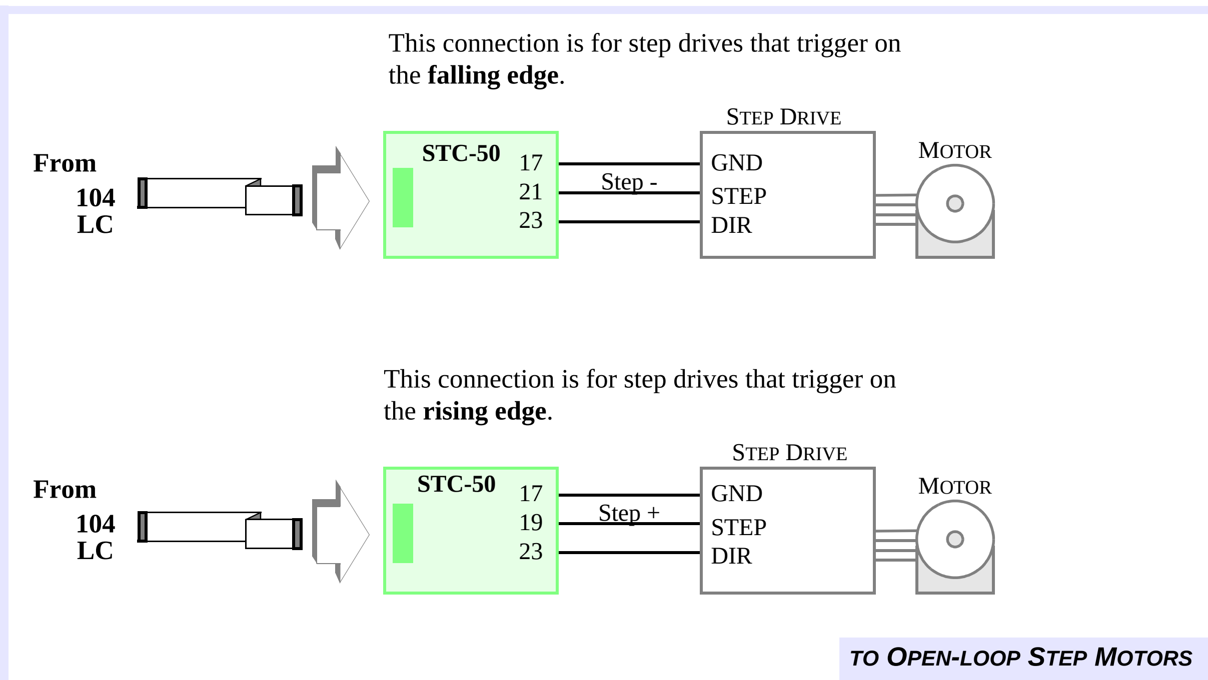
Most step drives require 3 wires for operation: step, direction and ground (or + 5V). The controller provides a TTL-level step pulse(+) output and direction(+) output for each axis. In addition, the complements of the step and direction are also provided (*Step-*, *Dir-*).

Some drives allow differential inputs in which both *Step+* and *Step-* lines are connected for higher noise immunity. If in doubt, fax the driver data sheets or driver pinouts to *Motion Engineering* along with any questions.

Note that when only *Step+* or *Step-* is used, it may be necessary *to jumper unused terminals on the step drive*. Before connecting *Step+* or *Step-*, consult your step drive's manual.

Note	If the controller is configured for *open loop step control*, make sure that the tuning parameters conform to the parameters listed in *Open-Loop Stepper Systems* (on page 6-7 in Chapter 6).

Figure 4-10 Typical Open-Loop Step Motor Connections



Closed-loop Step Motors

DSP Series controllers can control step motors with encoder feedback. Closed-loop steps are controlled by a PID algorithm running on the DSP in real time. The controllers accept TTL-level (0V to +5V, 40mA max) encoder input from either differential or single-ended encoders. Differential encoders are preferred due to their excellent noise immunity. The connections for a single-ended encoder are identical to a differential encoder except, nothing should be connected to channel A- and channel B-. (The A- and B- lines are pulled up internally to 2.5V).

Encoder signals are read in quadrature. Every line on the encoder will produce a rising edge and a falling edge on channels A+ and B+, which is interpreted by the DSP controller as 4 encoder counts.

Connecting closed-loop step motors to the controller is similar to servo motors, except that the step and direction lines are connected instead of the analog signal. The minimum connections are:

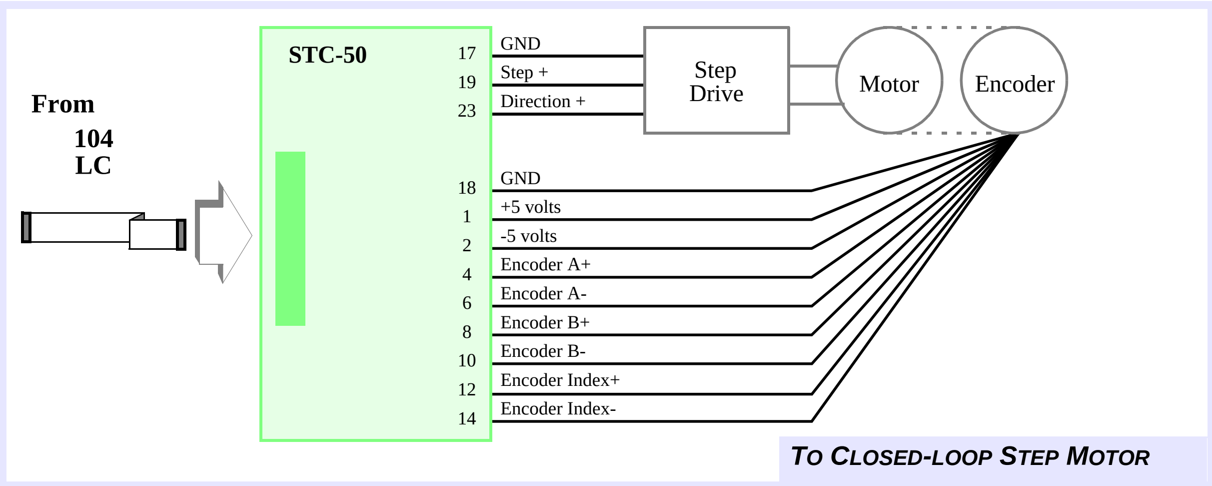
Step+ (or Step-)
Direction+ (or Direction-)
Signal Ground
Encoder A+ and B+ lines
+ 5V

Note that when only *Step+* or *Step-* is used, it may be necessary *to jumper unused terminals on the step drive*. Before connecting *Step+* or *Step-*, consult your step drive's manual

In general, use *Step+* for drives with active high logic, and use *Step-* for drives with active low logic. Both *Step+* and *Step-* lines can be connected to drives with differential inputs. If in doubt, fax the drive pinouts to Motion Engineering along with any questions.

Warning!	For the best performance, ensure that the ratio is between the *encoder resolution* (counts per revolution) and the *step resolution* (steps per microsteps per revolution) is 1:4. Lower ratios (1:1, 1:2) will be difficult to tune and will have poor static stability. Higher ratios (1:6, 1:8) will have poor constant velocity stability.

Figure 4-11 Typical Connections for Closed-Loop Step Motor



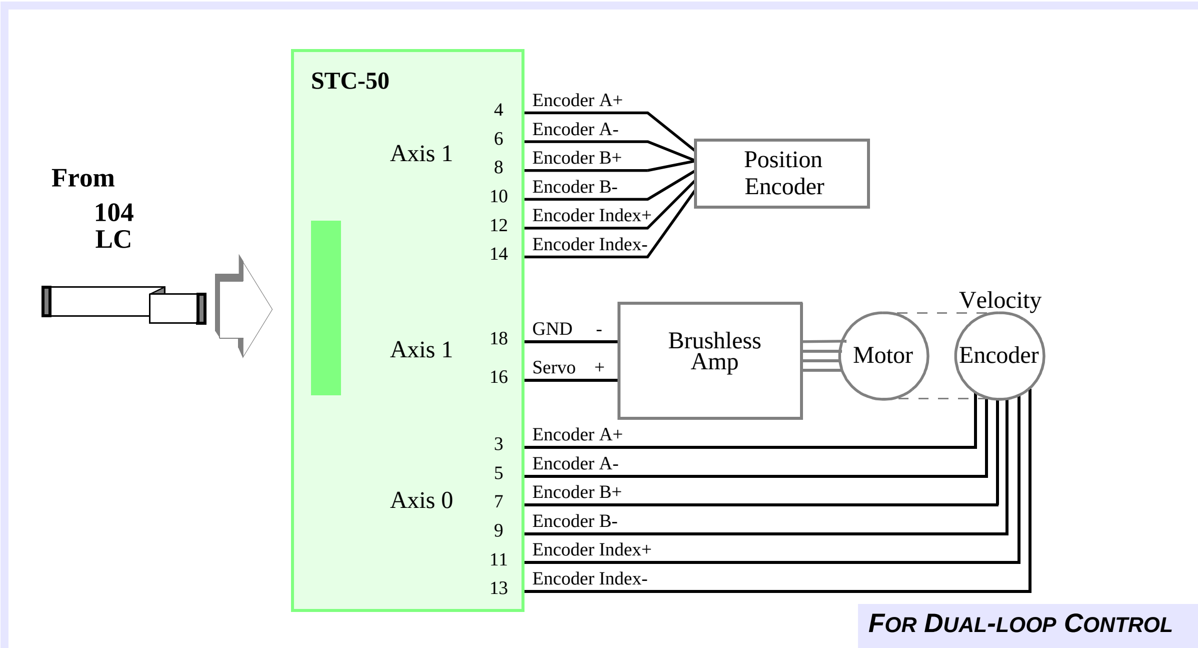
Note	For drives that trigger on the rising edge of the pulse input, use *Step+*. For drives that trigger on the falling edge of the pulse input, use *Step-*.

Connections for Dual-Loop Control

DSP Series controllers can be configured for dual-loop control. In dual-loop control, the velocity information for the PID derivative term (Kd) is derived from a rotary encoder on the motor shaft, and the position information for the PID proportional and integral terms are derived from an encoder on the load itself.

The axis that will be used for the rotary encoder is configurable through software and can be any axis that is not controlling a motor. For example, if axis 0 is configured for velocity feedback and axis 1 is configured for positional feedback, your system would be connected as shown in the next figure.

Figure 4-12 Typical Dual-loop Encoder Wiring with Differential Encoders



PCI

Connections to Servo Motors

PCI/DSP controllers can control brush servo motors, brushless servo motors, or linear brushless motors. Basic connections require an analog output signal (from the controller to the amplifier) and an encoder input (from the motor to the controller).

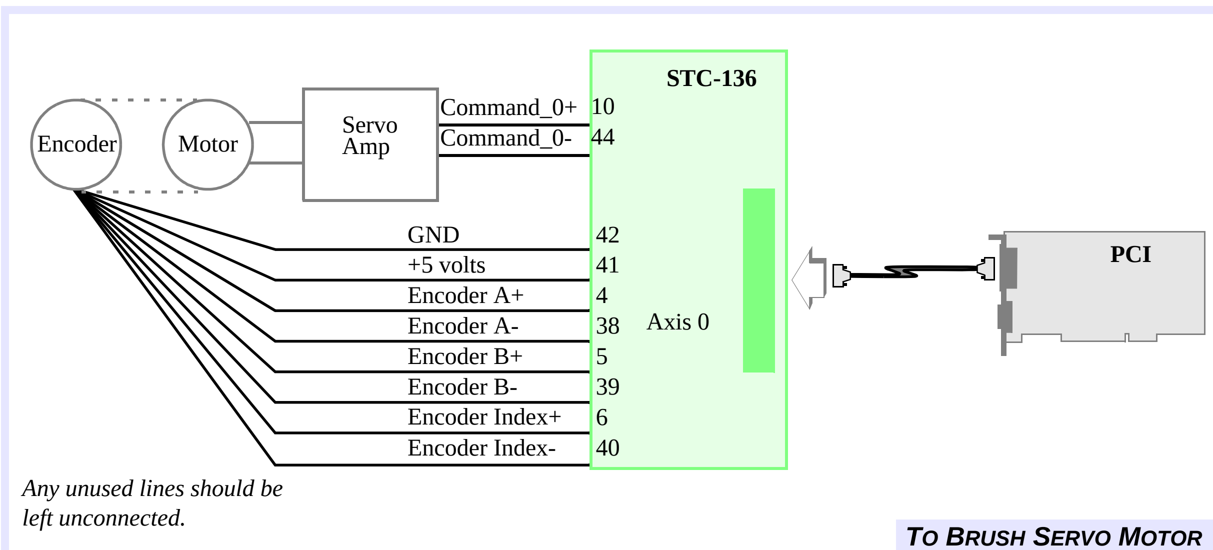
Most amplifiers support either Velocity mode (voltage control), Torque mode (current control) or both. The PCI controller can be used with either servo motor/amplifier package.

PCI controllers accept TTL-level (0V to +5V, 40mA max) encoder input from either differential or single-ended controllers. Differential encoders are preferred due to their excellent noise immunity. See Figure 4-19, *Typical Single-Ended Encoder Connections*, for instructions.

Brush Servo Motors

The minimum required connections to a brush-type servo are: Analog signal (+/- 10V), +5V, Signal Ground, Encoder Channel A+, Encoder Channel B+. Typical connections for a brush servo motor with differential encoder are:

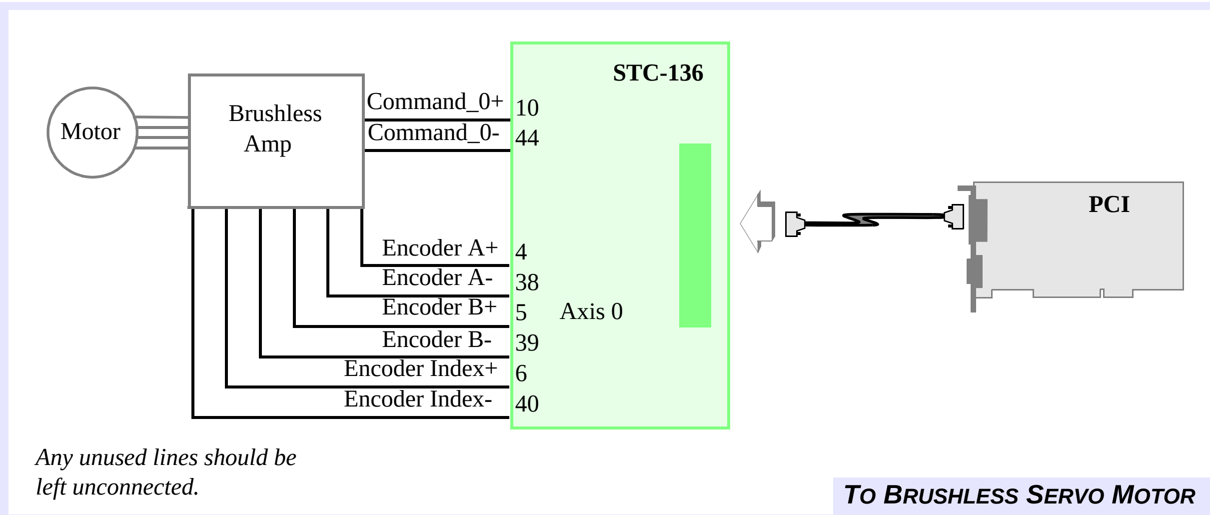
Figure 4-13 Typical Brush Servo Motor Connections



Brushless Servo Motors

Typical connections for a brushless servo motor with a differential encoder are:

Figure 4-14 Typical Brushless Servo Motor Connections



Step-and-Direction Controlled Servo Motors

Some brushless servos are controlled by step-and-direction pulses. With this scheme, the position information is communicated by step pulses, and the PID loop is handled internally by the drive itself.

To avoid possible instability caused by conflict between the drive PID loop and the controller PID loop, **operate step-and-direction servos as open-loop step motors.** The controller will send step pulses and a direction pulse to the drive, which will handle the PID position control loop internally.

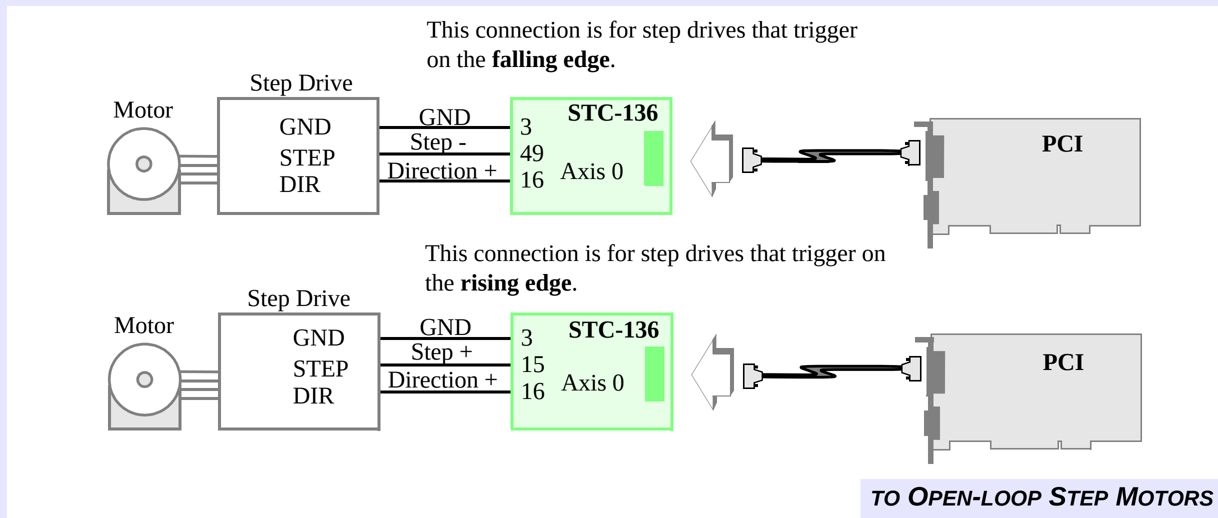
Warning!	If the controller is configured for open loop step control, make sure that the tuning parameters conform to those listed in *Open-Loop Stepper Systems* (Chapter 6).

Connections to Step Motors

Open-Loop Step Motors

The PCI controllers can control step motors in both open-loop (no encoder) and closed-loop configurations. In the open-loop configuration, the step pulse output (connected to the drive) is fed back internally and used to keep track of the "actual position." With open-loop step configuration selected, the DSP closed the loop internally on a pair of axes. PCI controllers are compatible with full/half and micro stepping drives.

Figure 4-15 Typical Open-Loop Step Motor Connections (PCI)



Closed-Loop Step Motors

PCI controllers can control step motors with encoder feedback. Closed-loop steps are controlled by a PID algorithm running on the DSP in real time. The controller's accept TTL-level (0V to 5V, 40mA max) encoder input from either differential or single-ended encoders. Differential encoders are preferred due to their excellent noise immunity.

Encoder signals are read in quadrature. Every line on the encoder produce a rising edge and a falling edge on channels A+ and B+ which is interpreted by the PCI controller as 4 encoder counts.

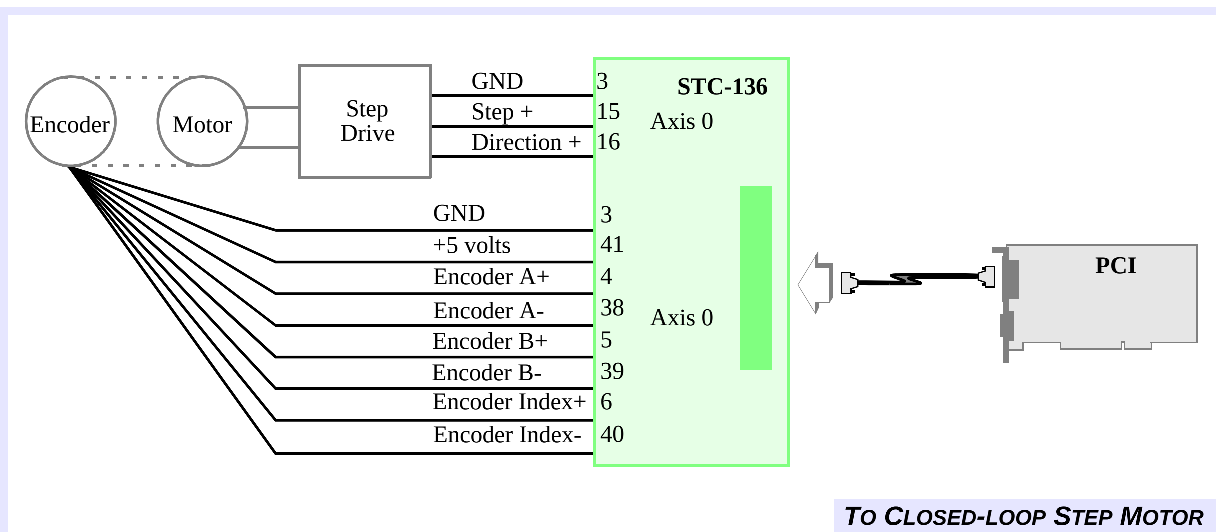
Connecting closed-loop step motors to the controller is similar to servo motors, except that the step and direction lines are connected instead of the analog signal. The minimum connections are Step+ (or Step-), Direction+ (or Direction-), Signal Ground, Encoder A+ and B+ lines, and +5V.

Note that when only *Step+* or *Step-* is used, it may be necessary to *jumper unused terminals on the step drive.* Before connecting *Step+* or *Step-*, consult your step drive's manual.

In general, use *Step+* for drives with active high logic, and use *Step-* for drives with active low logic. Both *Step+* and *Step-* lines can be connected to drives with differential inputs. If in doubt, fax the drive's pinouts to *Motion Engineering* along with any questions.

Warning!	For the best performance, ensure that the ratio between the *encoder resolution* (counts per revolution) and the *step resolution* (steps per microsteps per revolution) is 1:4. Lower ratios (1:1, 1:2) will be difficult to tune and will have poor static stability. Higher ratios (1:6, 1:8) will have poor constant velocity stability.

Figure 4-16 Typical Closed-Loop Step Motor Connections (PCI)



Note!	For drives that trigger on the rising edge of the pulse input, use *Step+*. For drives that trigger on the falling edge of the pulse input, use *Step-*.

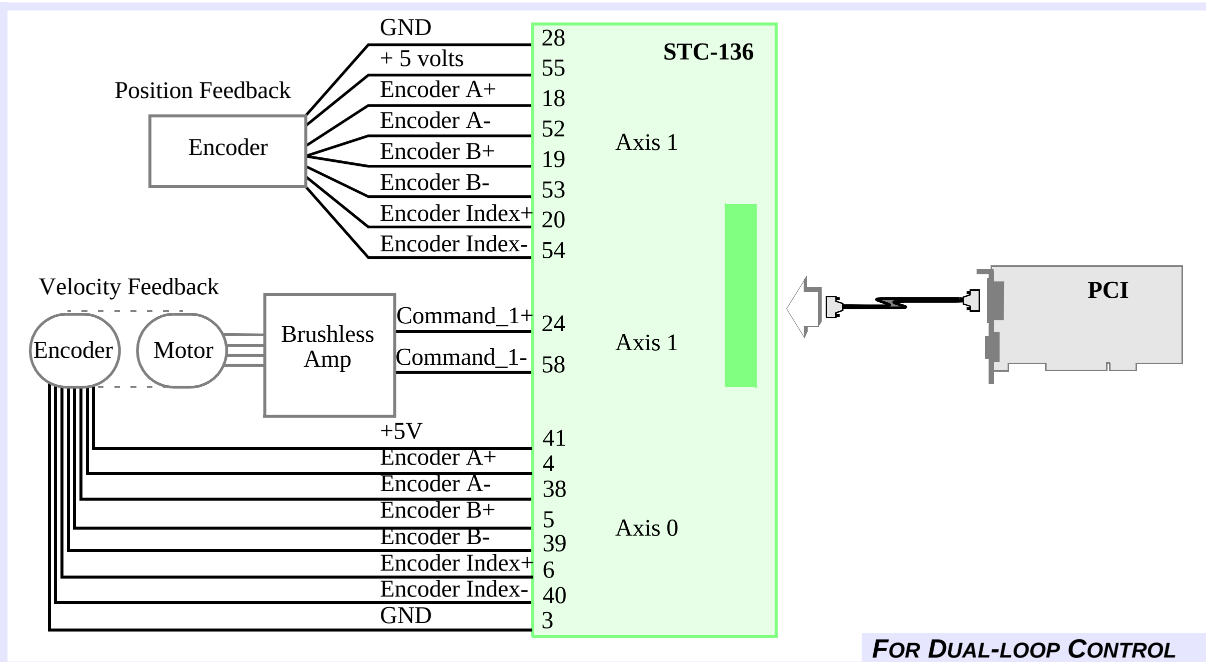
Connections for Dual-Loop Control

PCI controllers can be configured for dual-loop control. In dual-loop control, the *velocity information* for the PID derivative term (Kd) is typically derived from a rotary *encoder on the motor* shaft, and the *position information* for the PID proportional and integral terms is derived from an *encoder on the load* itself.

After the axes are configured for dual-loop control, all commanded motion & PID filter settings should be performed on the *position encoder* axis.

The axis that will be used for the velocity encoder is configurable through software and can be any axis that is not controlling a motor. For example, if axis 0 is configured for velocity feedback and axis 1 is configured for position feedback, your system would be connected as shown in the next figure:

Figure 4-17 Typical Dual-Loop Encoder Connections (PCI)



Connections for Encoder Signals

Differential encoders are preferred over single-ended encoders, because of their superior immunity to noise. There is one +5 volt supply and return shared by each pair of encoders, which is available at 2 sets of power pins (5V_OUT, GND) on each connector.

Figure 4-18 Typical Differntial Encoder Connections (PCI)

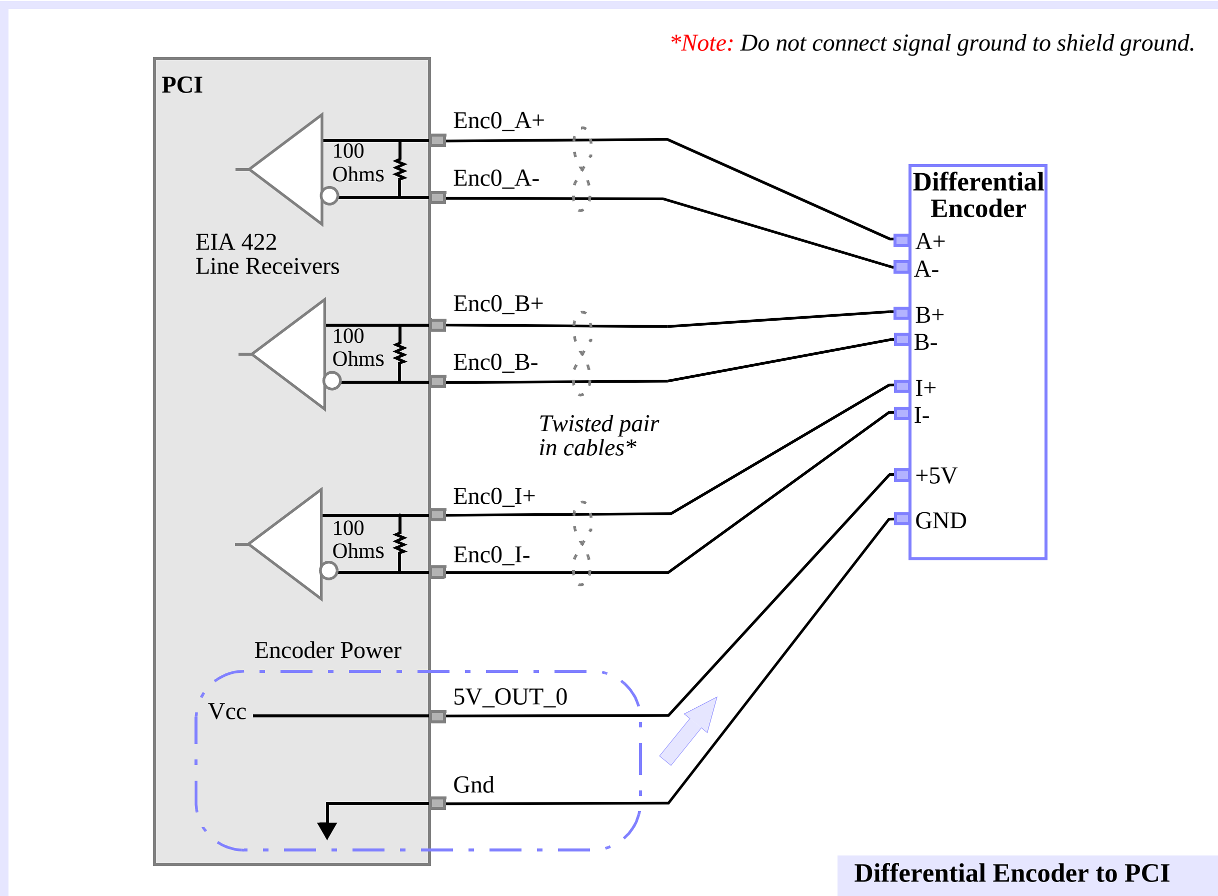
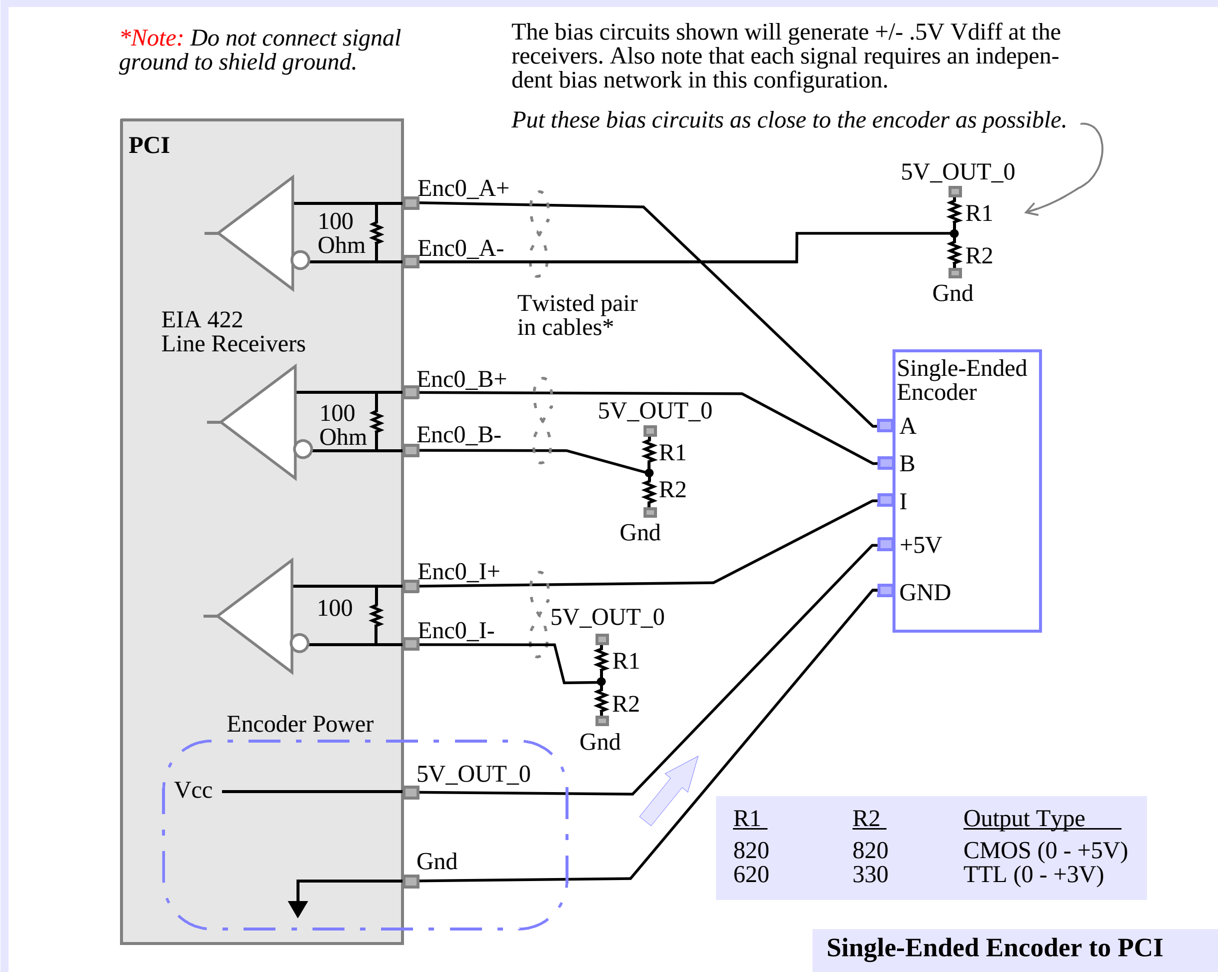


Figure 4-19 Typical Single-Ended Encoder Connections (PCI)



R1	R2	Output Type
820	820	CMOS (0 - +5V)
620	330	TTL (0 - +3V)

CHAPTER 5 CONNECT STCS TO DISCRETE I/O

Now make connections for the desired Dedicated and User I/O signals to the STC modules. After making those connections to the STC modules, **proceed to Chapter 6**, to test your system. For pinout information, refer to Appendix E, *Connections & Specifications*.

Dedicated and User I/O Notes

Opto-Isolation

(PCX, V6U, 104X, CPCI, STD only)

Dedicated and User I/O headers (connectors) conform to Opto-22/Grayhill/Gordos standard pin arrangement, and may be connected directly. Some Opto-22 racks do not use the +5V logic power on pin 49 of the I/O connector, and in those cases, +5V must be provided from an external source). Grayhill racks can be configured to take the +5V logic power from pin 49, so that no external source is necessary.

When the DSP Series controllers are powered up, the User I/O signals and Dedicated outputs come up *Low*. Most opto-isolation modules invert the I/O signals, which means that I/O signals may come up *High*. The active level of the Dedicated I/O signals can be configured in Motion Console; the boot configurations of the User I/O signals can be set using the function libraries. Refer to the *DSP Series C Programming Reference.*

Output Wiring

User I/O outputs are driven by an Intel 82C55 *Programmable Peripheral Interface Controller*. When power is supplied to the 82C55, these outputs have 3 possible output states:

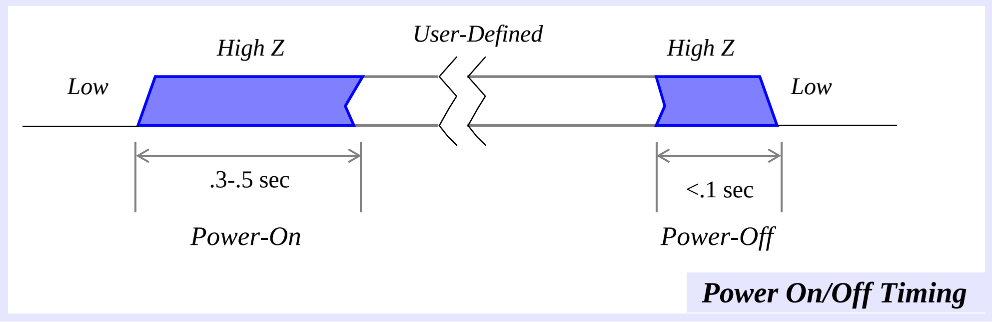
- **High Impedance** (High Z) (1 micro amp leakage current)
- **High** (>3.0V at 2.5 milliamp source current)
- **Low** (< 0.4V at 2.5 milliamperes sink current)

If there is no power to the 82C55, the output state is held low by input protection diodes.

The next figure shows the *power-on* and *power-off* timing of the controller output states. Approximately 0.3 to 0.5 seconds after power is supplied to the computer, the User outputs will go to the **Power-On** state. The **Power-On** state can be any one of the 3 output states of the 82C55 (High Z, High or Low).

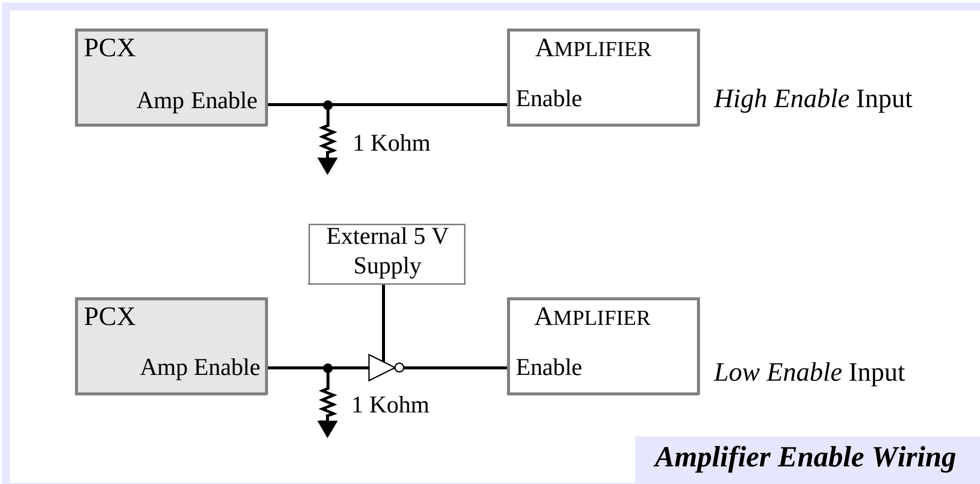
The **Power-On** state is configured at the factory to be the *Low* state.

Figure 5-1 Power On/Off Timing



For critical control signals that must always be in a defined state (such as amplifier enable/disable), your design should ensure that the default state of the 82C55 output is **Low**. You should use a pull-down resistor to insure that the output does not float high when the output is in the **High Z** impedance state. The next figure shows the correct wiring for amplifiers with *Low Enable* and *High Enable* inputs.

Figure 5-2 Amplifier Enable Wiring Using Pull-Down Resistors



Analog Input Wiring

(PCX, CPCI, STD, V6U Only)

Analog inputs are connected to the 20-pin connector P8. Pins 2 and 20 (Analog GND) are connected to the *logic ground* of the A/D chip **and** to a *separate ground plane* beneath the A/D chip. The *logic ground* of the A/D chip is also connected to the *bus ground* (with all of the other GND signals). When connecting analog inputs, use the separated *analog grounds* to improve noise immunity.

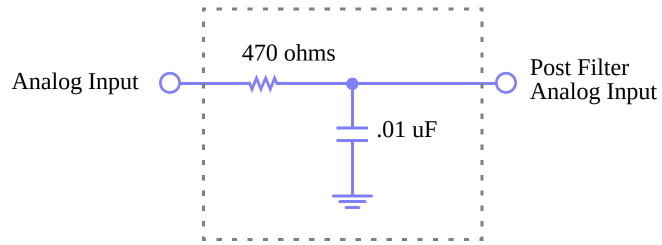
There are 8 channels, each with a 12-bit resolution. Each channel can be configured as either **Unipolar** (0 to +5V) or **Bipolar** (-2.5V to +2.5V). Because there is no buffer between the P8 connector and the actual A/D integrated circuit, the **input voltages must not exceed +5V or fall below -2.5V**.

Low Pass Filters on Analog Inputs (V6U only)

For Revision 4, we added low pass filters to each of the analog inputs, to prevent any unwanted noise from external sources.

Figure 5-3 V6U Analog Input Filters

A 34 kHz, single pole, low pass filter has been added to each of the analog inputs. The low pass filter anticipates source impedance of 50 ohms or less.



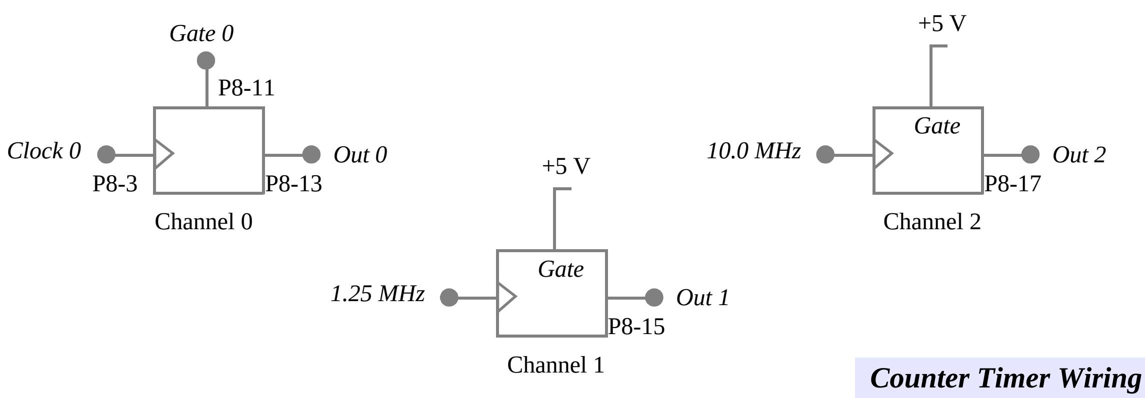
V6U Analog Input Filters

8254 Counter Wiring

(PCX, CPCI, STD, V6U Only)

There are 3 16-bit counters available for user functions. Counter 0 can accept an external clock input (pin 3 on P8) and Counters 1 and 2 have *fixed frequency inputs* of 1.25 and 10 MHz respectively. The gate signal for Counter 0 (used in some modes) is on pin 11 of P8. All counter outputs are available on P8.

Figure 5-4 Counter/Timer Wiring Diagram



Counter Timer Wiring

Home and Limit Switch Wiring

For small and electrically quiet machines, the home and limit switches can be wired directly to the dedicated inputs. For larger and more electrically noisier machines, we recommend using optical isolation. The following diagrams show the wiring for both types of machines.

Wiring Examples

Figure 5-5 Example Wiring Diagram for **Axis 0 Limit Switches** - Non-Opto-Isolated

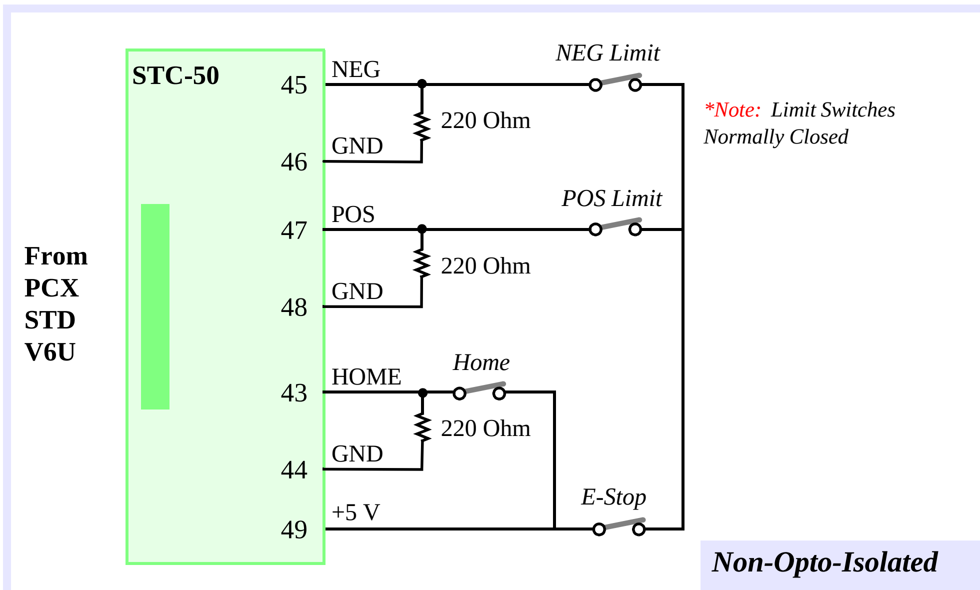


Figure 5-6 Example Wiring Diagram for **Axis 0 Limit Switches** - Opto-Isolated

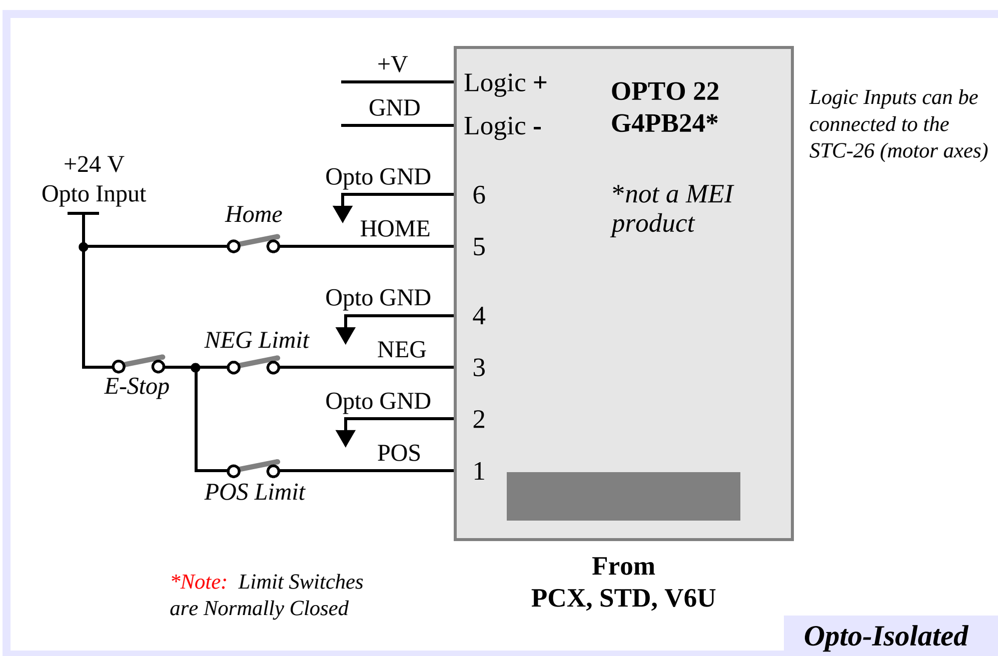
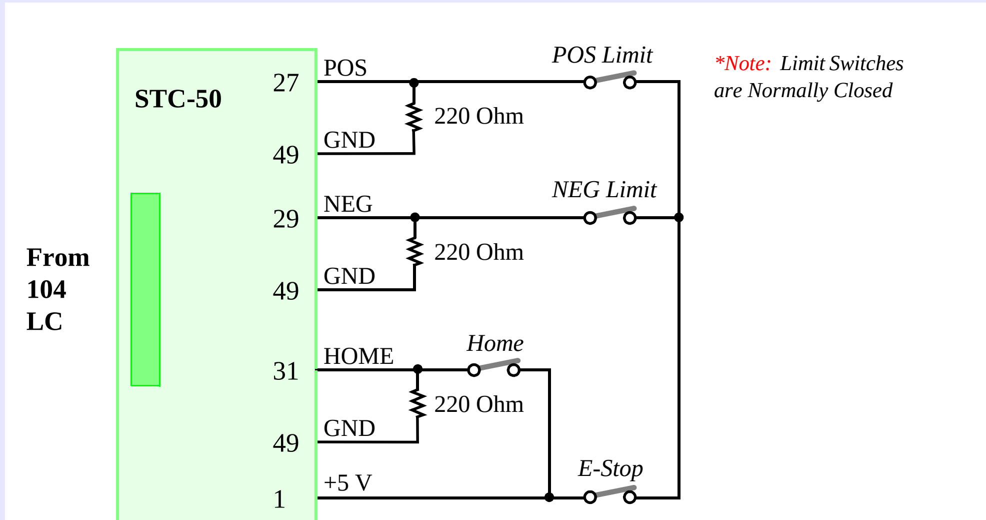


Figure 5-7 Example Wiring Diagram for **104 & LC Limit Switches** - Non Opto-Isolated



104 & LC Limit Switch - Non Opto-Isolated

Fail-safe limit operation is provided for both the optically isolated and non-isolated limit circuits. If a wire breaks in the limit circuit, the associated limit is activated and the motion is stopped until the problem is corrected. Since the controller can be configured for either active high or low inputs, other limit and home sensor circuits can be used.

For opto-isolation with the LC or 104, refer to *Appendix F, OptoCon Reference*.

PCI/DSP Connections

Opto-Isolation

The PCI controller contains Opto-Isolation for all the Discrete I/O except the In_Position bit. There are four Opto-inputs and one Opto-output per axis. There is an additional 24 lines of optically isolated, bi-directional User I/O. All I/O operates from 5-24 volts.

Warning!	Dedicated Outputs and User I/O require current limiting resistors

Opto-Circuit Specifications

Operating Temperature Range	0 - 60° C
User Voltage Range	24 V_{DC}

For Opto-inputs (Home*n*_IN, Pos_Lim*n*_IN, Neg_Lim*n*_IN, Amp_Flt*n*_IN)

Active Inputs Guaranteed	±3.5V max
Inactive Input Guaranteed	±1.0V max
Peak Operational Voltage	Vin = 45V max

For Opto-outputs (Amp_En*n*_C, Amp_En*n*_E)

Active Output Guaranteed	Iout = 10mA min Vout = .3V max
Inactive Output Guaranteed	Iout = .01mA max
Absolute Maximums (may damage parts if these are exceeded)	Iout = 50mA max Vout = 40V max Ireverse = 100mA max (Protection Diode)

For Opto-inputs (UserIO_*n*, where *n* is A, B, or C)

Active Input Guaranteed Max Input Voltage @ 2mA	1.7V
Inactive Input Guaranteed	Iin = .1mA max
Absolute Maximums (may damage parts if these are exceeded)	Iin = 50mA Vreverse = 40V (*see page 5-12, Bi-Directional User I/O*)

For Opto-outputs (UserIO_*n*, where *n* is A, B, or C)

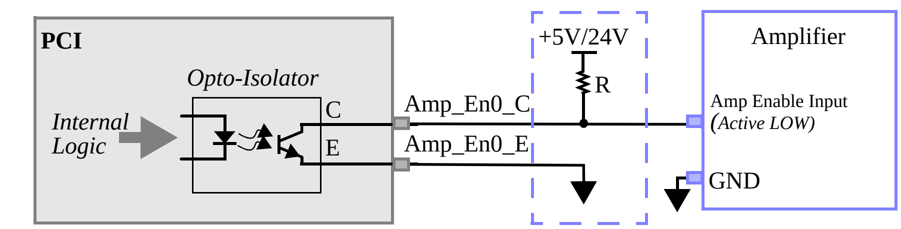
Active Output Guaranteed	Iout = 10mA min Vout = .3V max @ 10mA
Inactive Input Guaranteed	Iout = .01mA max
Absolute Maximums (may damage parts if these are exceeded)	Iout = 50mA Vout = 40V Ireverse = 50mA

Dedicated I/O - PCI

Output Wiring

Amplifier Enable Wiring

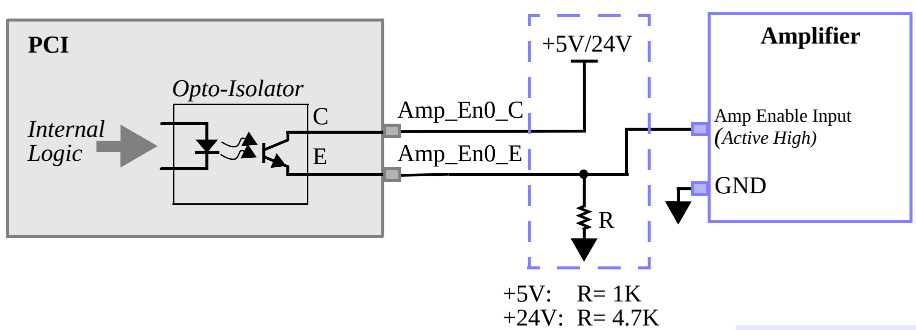
Figure 5-8 Example of Active Low Enable at Amp



Note: Verify that V_{CE} of the output is less than V_{IL} for the amplifier's enable input.

Active Low Enable at Amp

Figure 5-9 Example of Active High Enable at Amp

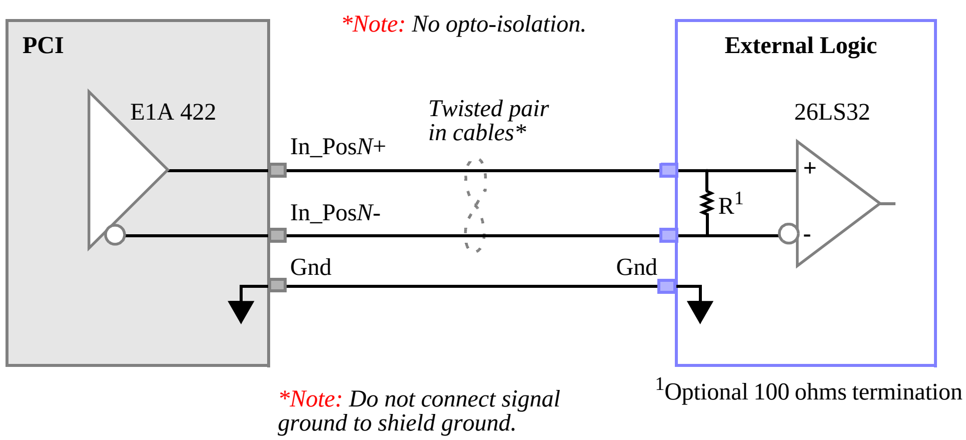


Active High Enable at Amp

In_Position Output Wiring

In_Position signals are differential EIA 422 outputs from the PCI. External logic that uses In_Pos/V signals should use a differential receiver such as the 26LS32.

Figure 5-10 Example In_Position Output Wiring



In_Position Outputs

Input Wiring

Amplifier Fault Input Wiring

Figure 5-11 Example of Pull-Up Logic

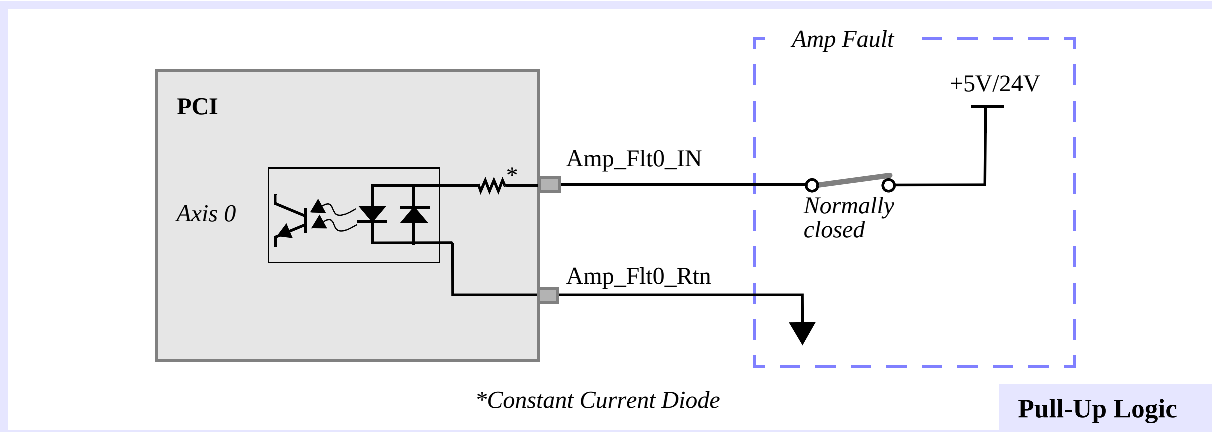
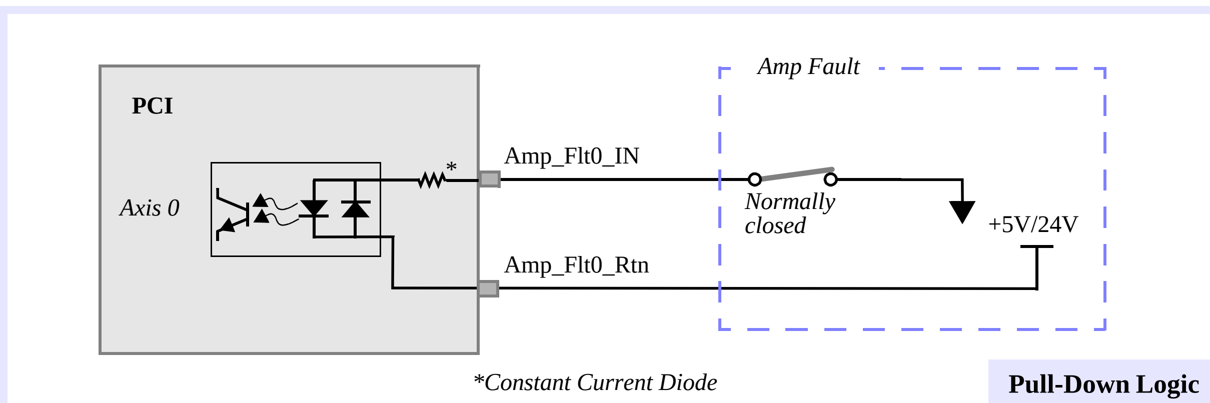


Figure 5-12 Example of Pull-Down Logic



Home and Limit Signals

Figure 5-13 Example of Common Gnd Logic

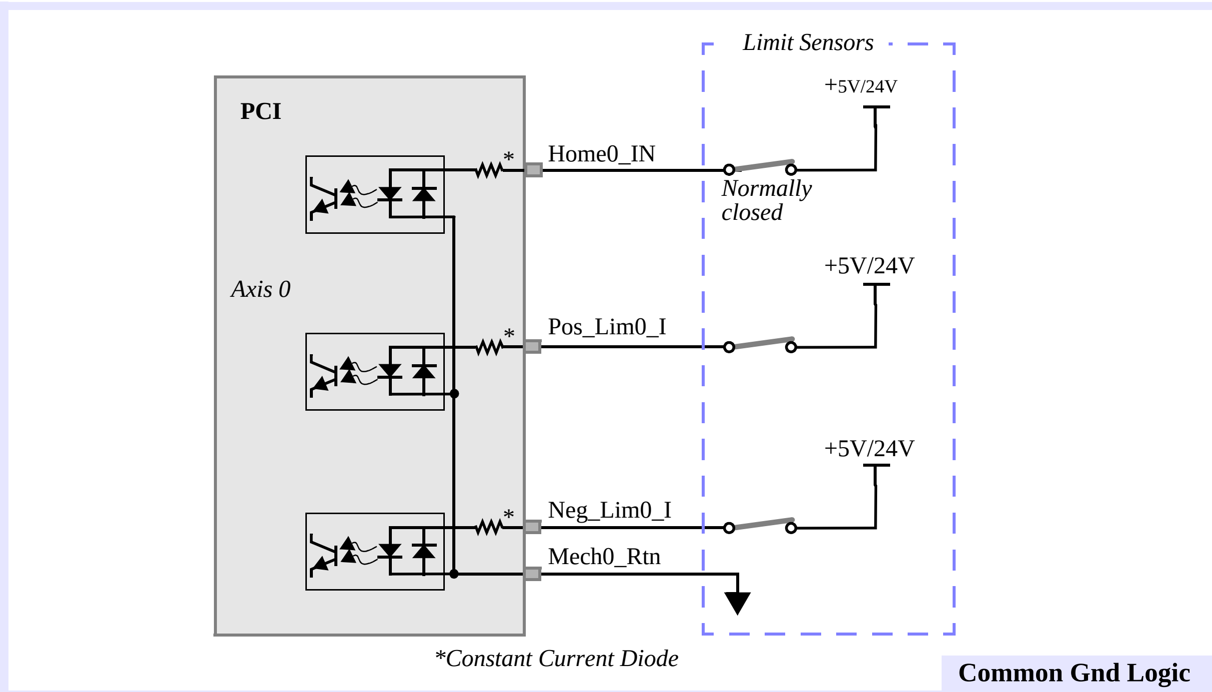
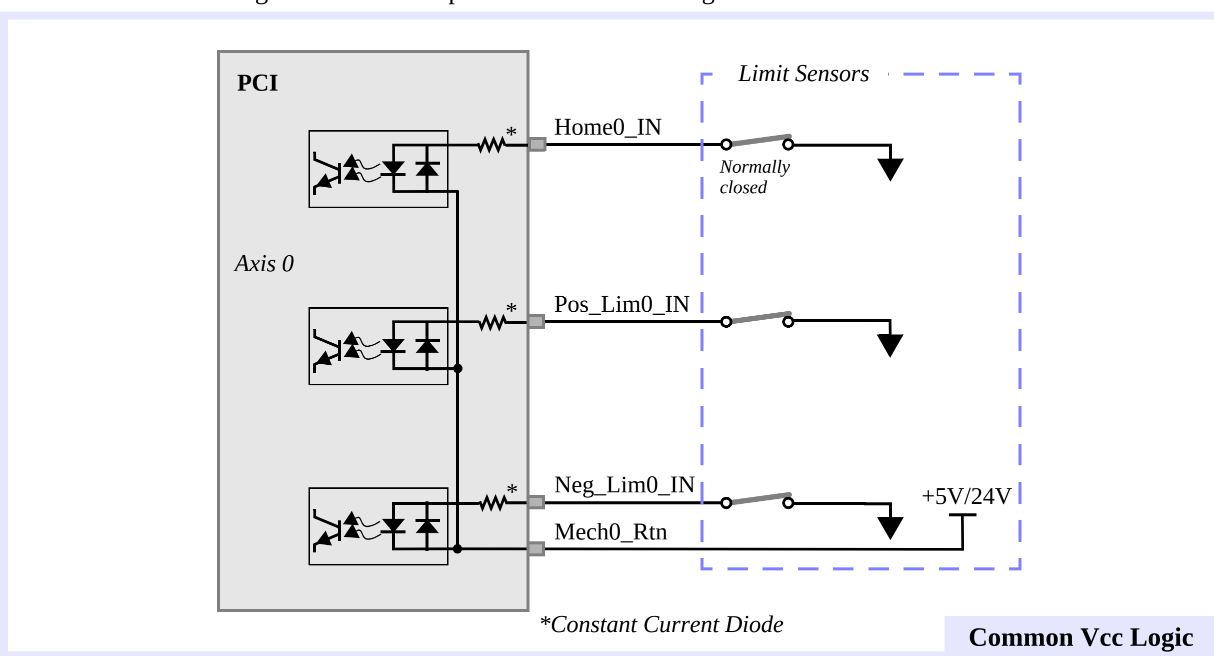


Figure 5-14 Example of Common Vcc Logic



Bi-Directional User I/O

Note:	To maintain electrical isolation between the PCI and external I/O, the power and ground connections should be from an external power source, and should not be tied to the PCI's power or ground connections.

Figure 5-15 Example of User I/O as Input

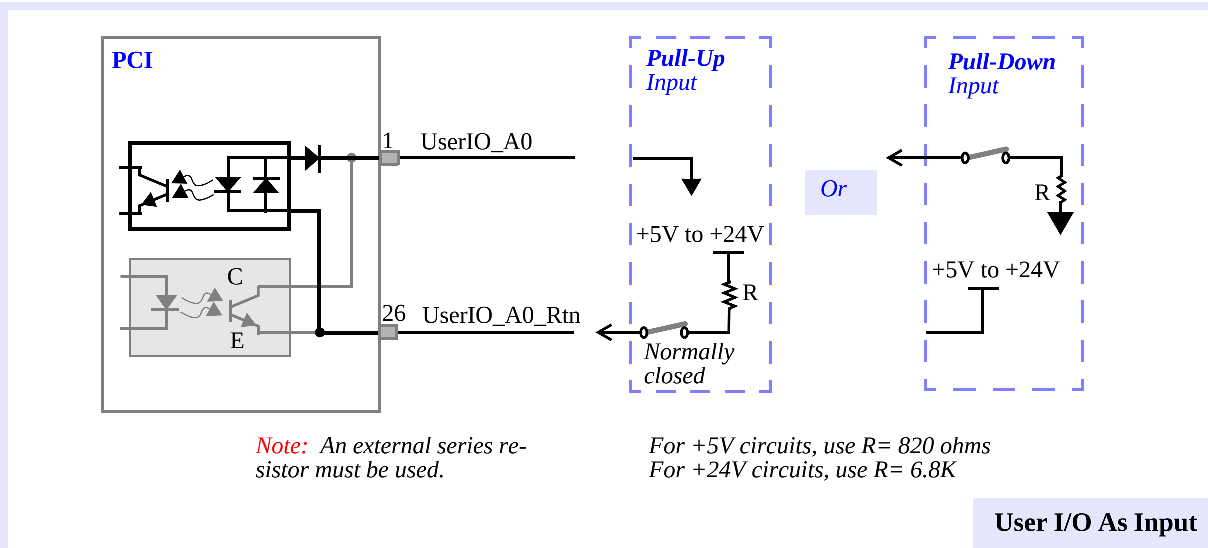
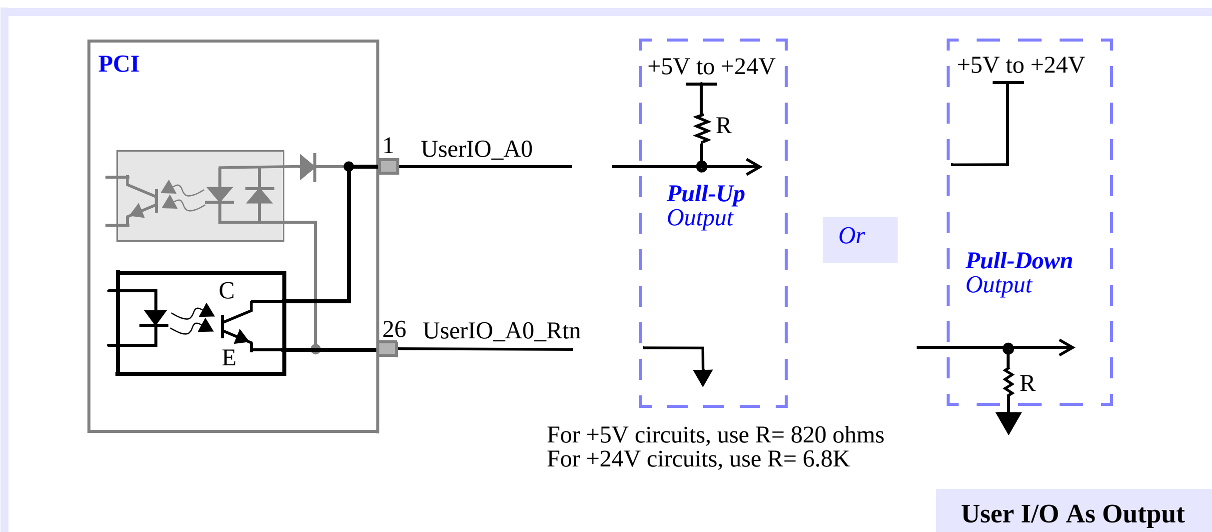


Figure 5-16 Example of User I/O as Output



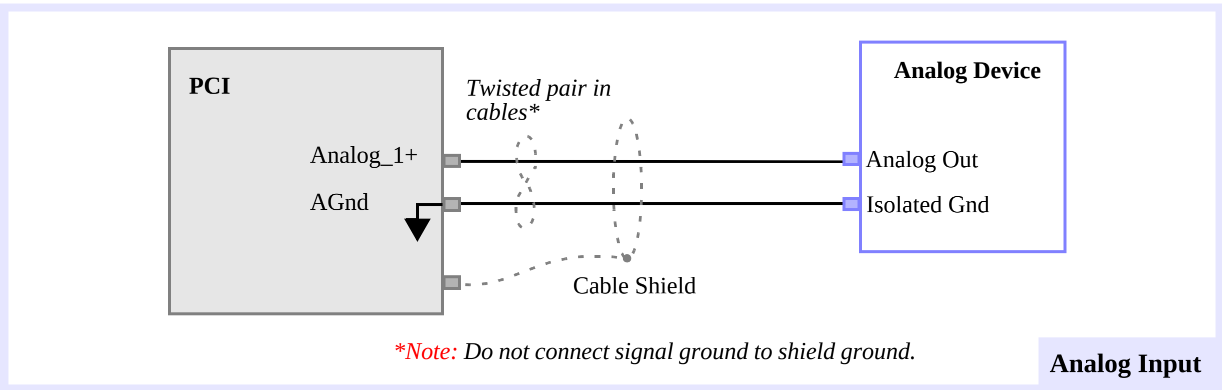
Analog Input Wiring

Pins 35, 36 and 67, 68 (Analog Gnd) are connected to the *logic ground* of the A/D chip **and** to a *separate ground plane* beneath the A/D chip. The *logic ground* of the A/D chip is also connected to the *bus ground* (with all of the other GND signals). When connecting analog inputs, use the separated *analog grounds* to improve noise immunity.

There are 8 channels, each with a 12-bit resolution. Each channel can be configured as either **Unipolar** (0 to +5V) or **Bipolar** (-2.5V to +2.5V). Because there is no buffer between the connector and the actual A/D integrated circuit, the **input voltages must not exceed +5V or fall below -2.5V.**

Use this configuration for an isolated analog source, such as a thermocouple:

Figure 5-17 Example of Analog input for an isolated analog source



CHAPTER 6 TEST SYSTEM

Closed-Loop Systems

To test servo motors and closed-loop step motors:

1. Connect the encoder.
2. Test the encoder connections: watch the *Actual* field in the *Axis Operation* window change while turning the motor shaft by hand.
3. Connect the motor. Choose the axis and click the *Abort* button in the *Axis Operation* window to disable PID control.
4. Manually turn the motor using the *Offset* field in the *Axis Operation* window.
5. Verify the motor/encoder phasing using the *Actual* field in the *Axis Operation* window.
6. Exercise and tune the PID control loop.

We recommend testing the wiring of closed-loop systems at each step. This method should make the process easier and save time.

This procedure assumes that you have successfully installed the controller, and that the Motion Console program can execute properly. Also, before testing your system, you must configure the *Axis Configuration* property page for closed-loop operation and select the appropriate motor type.

Step 1: Connect Encoder

Turn off the computer. Attach all encoder leads according to the manufacturer's wiring diagram and the instructions provided in this manual.

WARNING!	**Do not attach the motor signal wires yet!**

Turn on the computer. Note that the controller provides the +5V power (which comes directly from the host computer's power supply) to the encoder for most brush servo and step motor systems.

If the servo motor uses the encoder for commutation and the servo amplifier provides the encoder power, the servo amp **must be turned on** to test the encoders.

Step 2: Test Encoder Connections

Start the Motion Console program. Choose an axis in the Hardware Summary window and click the *Configure Axis* button. In the *Axis Configuration* property page verify that the axis' configuration is accurate for your system. Close the property page and open the *Axis Operation* window which will display the actual encoder position.

Turn the motor shaft/encoder by hand. The counts in the *Actual* field should increase and decrease normally. Check to see that 1 revolution of the encoder provides the correct number of encoder counts (number of encoder lines x 4).

Tip!	
Encoder Counts Bounce	If the encoder counts "bounce" by one count when the motor shaft is turned (i.e. change up and down one count when the encoder is tuned), the likely problem is that the one side of the encoder (A or B) is not connected. Check the connections carefully.

Step 3: Connect the Motor

Turn off the power to the computer.

WARNING!	**Be sure the power to the servo amp/step drive is off!**

Connect the analog motor command or step/direction lines. **Turn the computer power on.**

Step 4: Manually Turn the Motor

Click the *Abort* button in the *Axis Operation* window to disable PID control. Turn on power for the servo amp/step drive. The shaft of the servo motor should now turn freely (for torque mode amplifiers).

Enter a value (**10**) in the *Offset* field of the *Tuning Parameters* display (still in the *Axis Operation* window) to start turning the motor. Increase the *Offset* value past 10 until the motor begins to turn slowly.

If the motor does not turn with approximately **1000** counts of offset, check the output of the controller with a voltmeter. Note that the *Offset* field range is +/-32,767 counts, corresponding to +/-10V or +/- full scale step output.

Tip! *Motor Doesn't Turn*	If the motor will not turn when an offset is applied, check the motor and amplifier connections, and also check that the *State* field reads *Abort Event*, to make sure the PID control is disabled.

Next, disconnect the amplifier connections to the controller and use a voltmeter to verify that the controller *is* outputting a motor signal. Remember that the voltmeter will at best pick up an average value for the step output.

Note that +/-32,767 counts, corresponds to +/-10V and +/- full scale step output. If no voltage is present, contact Motion Engineering for assistance.

Step 5: Verify Motor/Encoder Phasing

With the motor turning slowly under a manually applied offset, check the *Actual* position field in the *Axis Operation* window to see if the encoder counts are increasing or decreasing.

Table 6-1 Correct Motor/Encoder Phasing

Offset	***Encoder Counts***	***Phasing***
Positive (+ Value)	Increasing	Correct
Positive (+ Value)	Decreasing	Wrong
Negative (- Value)	Decreasing	Correct
Negative (- Value)	Increasing	Wrong

If the phasing is incorrect, set the offset to zero, turn off the servo amplifier/step drive and the host computer, and swap the A and B leads (A+ for B+ and A- for B-) to the encoder. Then repeat Steps 4-5 to verify proper motor/encoder phasing.

Step 6: Exercise the System

Setting the tuning parameters is part science and part art. Closed loop performance depends on the tuning parameters, servo amp/step drive, and the mechanical system. Finding optimum tuning parameters requires experimentation, theoretical understanding of PID control loops, and practical experience.

Tip! *Tune It TWICE*	We highly recommend tuning the system **twice**. First tune the system **with the motor disconnected from the mechanical system**, to gain familiarity with the procedure. Second, **connect the motor to the mechanical system** and re-tune.

Before tuning, verify the settings for the axis in the *Axis Configuration* property page. Double-click on the axis to open the *Axis Operation* window and refer to the *Tuning Parameters* display. Start with the parameters in the next table.

Table 6-2 Tuning Parameters

Parameter	***Servos***	***Closed-Loop Steppers***
Proportional (Kp)	100	20 (Depends on step/encoder pulse ratio)
Integral (Ki)	2	0
Derivative (Kd)	400	0
Accel FF	0	0
Vel FF	0	1000 (Depends on step/encoder pulse ratio)
Integ. Max	32767	100
Offset	0	0
Limit	3500	3500
Scale	-5	-1 (slow), -3 (med), -5 (fast), -6 (superfast)
Friction FF	0	0

Note the setting for output limit. A value of 3500 will limit the voltage output to approximately 1V or 10% of full-scale step speed. In case a runaway occurs, the low setting will limit the power of a servo motor and the speed of a step.

Click the *Clear Positions* button in the *Position Status* display. Click the *Clear Fault* button in the *Axis Status* display. The servo motor's shaft should offer resistance when turned by hand.

Tip! *Motor Runs Away*	If the motor begins to "run away" without stopping when the shaft is turned by hand, it is likely that the encoder and motor are both out of phase. Turn off the power and swap the encoder A and B leads (both + and - on a differential encoder) and repeat the test.

Enter values in the *Position 1*, *Velocity*, and *Acceleration* fields to command motion. If the motor turns, proceed to tuning the system. If the motor does not turn, re-check each step.

Tip! *Motor Doesn't Turn*	If the motor fails to turn during exercising, check the *State* field for the software limits, E-stops, or other error conditions. Also, click the *Clear Position* button in the *Position Status* display to clear position.

Note that the default "in-position" window is 100 encoder counts. If, while in repeat mode, a move fails to reach the final position within that range, a second motion will not be initiated.

Step 7: Tune the System

Use the arrow buttons (← for *Position 1* and → for *Position 2*) in the *Movement* controls to start motion. If the motor begins to move back-and-forth, proceed to tuning. If the motor fails to turn, recheck each step.

Once the point-to-point motion can be commanded, the system can be tuned. See Appendix D, *Tuning Closed-Loop Systems*, for tuning concepts and a step-by step procedure for tuning closed-loop systems.

The primary tools used in tuning closed-loop systems are fields in the *Movement*, *Motion Parameters*, and *Position Status* sections and also the *Motion Graph* window (these are all described in Appendix B, *Motion Console*).

Use the fields in the *Movement* and *Motion Parameters* controls to initiate point-to-point motion in trapezoidal profile mode. Suggested settings for initial exercising are:

Table 6-3 Tuning Parameters for Closed-Loop Systems

Parameter	***Value***
Delay	1
Position 1	0
Position 2	4000*
Velocity	500
Acceleration	500

**or the number of encoder counts corresponding to one motor revolution*

Tip! *Motor Doesn't Turn*	If the motor fails to turn during exercising, check the *State* field window for software limits, E-stops, or other error conditions. Also, click the *Clear Position* button in the *Position Status* display to clear position.

Note that the default "in-position" window is 100 encoder counts. If, while in repeat mode, a move fails to reach the final position within that range, a second motion will not be initiated.

The fields in the *Position Status* display show the command and actual position, velocity, acceleration and position error of the axis in real time.

To view a plot of the motion, enter motion values in the *Movement* and *Motion Parameters* fields, click *Repeat Mode* on and start the motion with the arrow key. Click the *Motion Graph* button. In the *Motion Graph* window, select *Continuous* or *Sampled* and choose the parameter you want to graph (position, voltage, velocity, or error).

Open-Loop Stepper Systems

To test an open-loop stepper system:

Step 1: Connect the step drive.

Step 2: Manually turn the motor using the *Offset* field in the *Axis Operation* window.

Step 3: Exercise the motor.

WARNING!	**Always disconnect the motor shaft from the machine when testing connections or software.**

This procedure assumes that you have successfully installed the controller, and that Motion Console program can execute properly.

Before testing your open-loop stepper system, you must configure the *Axis Configuration* property page for open-loop operation and a step motor type, and also select an appropriate speed range.

Step 1: Connect Wires

Turn off the computer. Connect the wires to the step drive as shown in this manual, or as shown in the step drive manual.

Step 2: Manually Turn the Motor

Choose the axis and open the *Axis Operation* window. Click the *Abort* button to disable PID control. Click *Enable* in the *Amplifier* group. Enter a value (**10**) in the *Offset* field of the *Tuning Parameters* controls. Increase the Offset until the motor begins to turn slowly.

If the motor does not turn with approximately **1000** counts of offset, check the output of the controller with a voltmeter. Note that the Offset range is +/-32,767 counts, corresponding to +/-10V or +/- full scale step output.

Tip! *Motor Doesn't Turn*	If the motor will not turn when an offset is applied, check the motor and amplifier connections, and also check that the *State* field reads *Abort Event*, to make sure that the motor is idle.

Next, disconnect the amplifier connections to the controller and use a voltmeter to verify that the controller is outputting a motor signal. Remember that the voltmeter will at best pick up a average value for the step output.

Note that +/-32,767 counts corresponds to +/-10V and +/- full scale step output. If no voltage is present, contact Motion Engineering for assistance.

Step 3: Exercise the Motor

For each axis configured for open-loop step motors, use the values listed in the next table for the *Tuning Parameters* controls. The *Scale* parameter changes accordingly to the speed range selected in the *Axis Configuration* property page.

Table 6-4 Tuning Parameters for Open-Loop Steppers

Parameter	***Value***
Proportional (Kp)	320
Integral (Ki)	32
Derivative (Kd)	0
Accel FF	32
Vel FF	3750
Integ Max	32767
Offset	0
Limit	32767
Scale	-1 (slow), -3 (med), -5 (fast), -6 (superfast)

Use the fields in the *Movement* and *Motion Parameters* controls to command point-to-point motion.

APPENDIX A MORE ABOUT WIRING

Wiring Servo Motors

DSP Series controllers can control brush servo motors, brushless servo motors, or linear brush/brushless motors. Basic connections require an analog output signal (from the controller to the amplifier) and an encoder input (from the motor to the controller).

Velocity/Torque Mode

Most amplifiers support either Velocity mode (voltage control), Torque mode (current control) or both. The DSP controller can be used with either servo motor/amplifier package. Generally, velocity mode is more stable than torque mode.

When the amplifier is in Velocity mode, the velocity of the motor is proportional to the analog input voltage (-10V to +10V). When the amplifier is in Torque mode the current applied to the motor is proportional to the analog input voltage (-10V to +10V).

Mode			
Velocity	**velocity** of motor	***is proportional to***	analog input voltage
Torque	**current** applied to motor		(-10V to +10V)

Encoder Input

DSP Series controllers accept TTL-level (0V to +5V, 40mA max) encoder input from either differential or single-ended encoders. Differential encoders are preferred due to their excellent noise immunity. When used with differential encoders, the differential line receiver on the controller reads the difference between A+ and A- and between B+ and B-. By reading the difference between the square wave inputs any significant noise is canceled out.

The connections for a single-ended encoder are identical to a differential encoder except that no connections are made to channel A- and channel B-. (The A- and B- lines are pulled up internally to 2.5V).

The controller reads the index pulse (either single-ended or differential ended). Typically, there is one index pulse per revolution of the encoder (rotary type), which can be used for homing.

Encoder signals are read in quadrature. Every line on the encoder will produce a rising edge and a falling edge on channels A+ and B+, which are interpreted by the DSP controller as 4 encoder counts.

Brush/Brushless Servo Motors

The minimum required connections to brush type servo are:

Analog signal (+/- 10V)
Signal Ground
Encoder Channel A+
Encoder Channel B+
+5V

Any unused lines should be left unconnected.

Step-and-Direction Controlled Servo Motors

Some brushless servos are controlled by step-and-direction pulses. With this scheme, the position information is communicated by step pulses, and the PID loop is handled internally by the drive itself.

Step-and-Direction servo systems can be operated in open-loop or closed-loop controller configurations. To avoid possible instability caused by conflict between the *drive* PID loop and the *controller's* PID loop, you should operate step-and-direction servos as open-loop step motors. (The controller will send step pulses and a direction pulse to the drive, which will handle the PID internally.) Generally, the best performance occurs when the controller is configured for open-loop steppers.

Warning!	If the controller is configured for *open loop* step control, make sure that the tuning parameters conform to those listed in *Test System: Open-Loop Stepper Systems* (page 6-7, Chapter 6).

When configured for open-loop steppers, the controller sends step and direction position information to the drive. The drive closes the torque, velocity, and position loops internally.

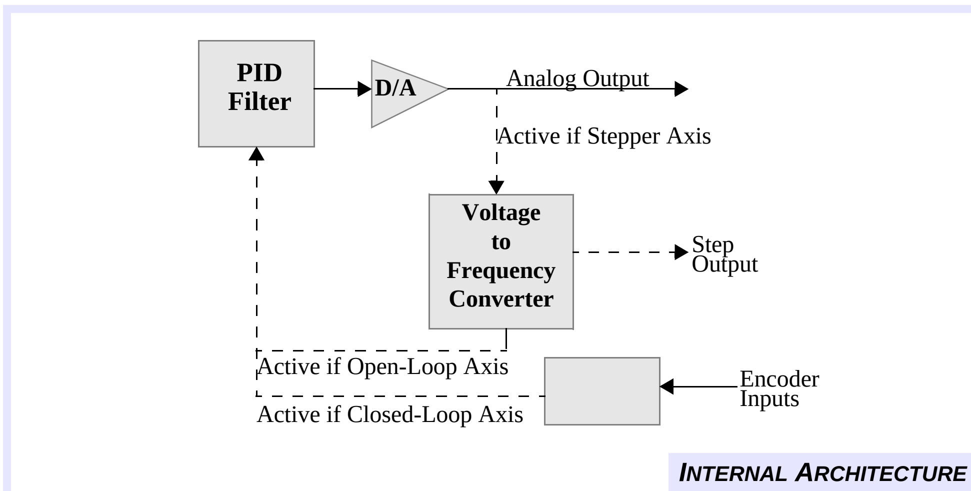
When configured for closed-loop steppers, the controller sends step and direction position information to the drive and receives action position information from the encoder. The drive closes the torque and velocity loops and the controller closes the position loop.

Wiring Step Motors

Open-Loop Step Motors

The DSP controllers can control step motors in both open-loop (no encoder) and closed-loop configurations. In the open-loop configuration the step pulse output (connected to the driver) is fed back into the line receivers and used to keep track of the "actual position." With open-loop step configuration selected, the DSP closes the loop internally on a pair of axes. Full/half and micro stepping drives are compatible with the boards.

Figure A-1 Internal Architecture to Control Step Motors



Most step drives require 3 wires for operation: step, direction and ground (or + 5V). The controller provides a TTL-level step pulse(+) output and direction(+) output for each axis. In addition, the complements of the step and direction are also provided (*Step*- and *Dir*-). Some drives allow differential inputs in which both *Step*+ and *Step*- lines are connected for higher noise immunity. If in doubt, fax the driver data sheets or driver pinouts to *Motion Engineering* along with any questions.

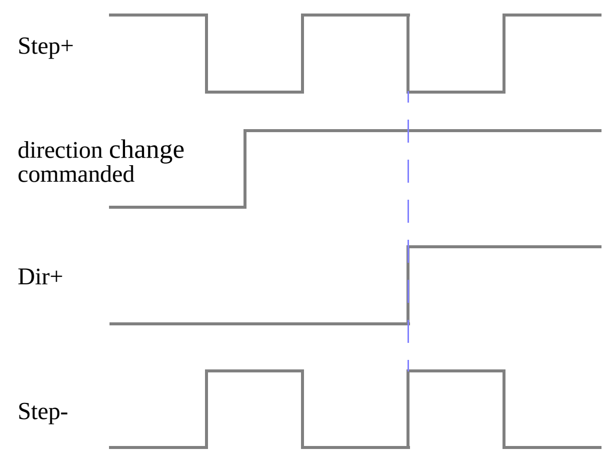
Note that when only *Step*+ or *Step*- is used, it may be necessary to jumper unused terminals on the step drive. Before connecting the step inputs, consult your step drive's manual.

Important!	For a listing of the tuning parameters required for motion with open-loop steps, refer to *Test System: Open-Loop Stepper Systems* (page 6-7, Chapter 6)

Direction Pulse Synchronization

The DSP Series controllers synchronize the direction pulse with the falling edge of the positive step pulse output. When connected to the step drive properly, it ensures that a step pulse and direction change will never occur at the same time.

Figure A-2 Direction Pulse Synchronization



DIRECTION PULSE SYNCHRONIZATION

Most step drives count pulses on either the rising edge or falling edge of the step pulse input.

If the Driver triggers on the	*Then*
falling edge	connect the controller's ***Step-*** to the pulse input on the drive
rising edge	connect the controller's ***Step+*** to the pulse input on the drive

The Direction(+) should be connected to the direction input of the drive. This guarantees that the drive will never receive a direction change during a step pulse.

Closed-Loop Step Motors

DSP Series controllers can control step motors with encoder feedback. Closed-loop steps are controlled by a PID algorithm running on the DSP in real time. The controllers accept TTL-level (0V to +5V, 40mA max) encoder input from either differential or single-ended encoders. Differential encoders are preferred due to their excellent noise immunity.

The connections for a single-ended encoder are identical to a differential encoder except that no connections are made to channel A- and channel B-. The A- and B- lines are pulled up internally to 2.5V.

Encoder signals are read in quadrature. Every line on the encoder will produce a rising edge and a falling edge on channels A+ and B+, which is interpreted by the DSP controller as 4 encoder counts.

Connecting closed-loop step motors to the controller is similar to servo motors, except that the step and direction lines are connected instead of the analog signal. The minimum connections are:

Step+ (or *Step-*)
Direction+ (or Direction-)
Signal Ground
Encoder A+ and B+ lines
+ 5V

Note that when only *Step+* or *Step-* is used, it is often necessary to **jumper unused terminals** on the step drive. Before connecting the step inputs, consult your step drive's manual.

In general, use *Step+* for drives with active high logic, and use *Step-* for drives with active low logic. Both *Step+* and *Step-* lines can be connected to drives with differential inputs. If in doubt, fax the drive's pin-outs to Motion Engineering, along with any questions.

APPENDIX B MOTION CONSOLE REFERENCE

Motion Console Supports all MEI DSP Series and SERCOS controllers and enables you to:

- Access and configure multiple controllers and their axes
- Configure Dedicated and User I/O lines
- Read axis status
- Upload/download firmware
- Tune your system using motion and tuning parameters
- Experiment with both absolute and relative motion, with repeat option
- Graph position, velocity, position error, and voltage for tuning, system diagnostics and analysis

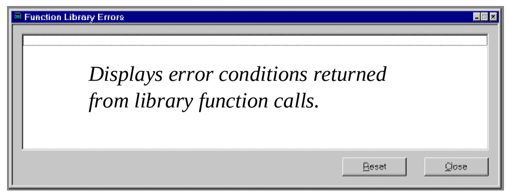
To run Motion Console you need one of these operating systems:

- Windows NT or Windows 95/98
- Windows 3.x with Win32S extensions (Motion Console will only run under 32-bit Window operating systems. Windows 3.11can be upgraded to Windows 32S in order to run Motion Console. The Windows 32S upgrade is available from Microsoft at no charge).

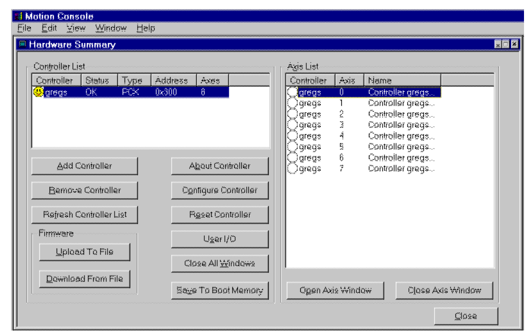
Motion Console is not designed to run under DOS. DOS users should use the DOS utilities provided by MEI (SETUP.EXE, CONFIG.EXE, VERSION.EXE).

Motion Console Windows

Library Errors

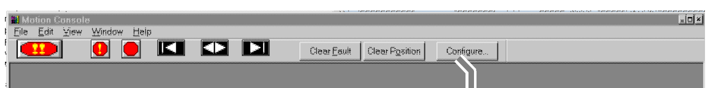


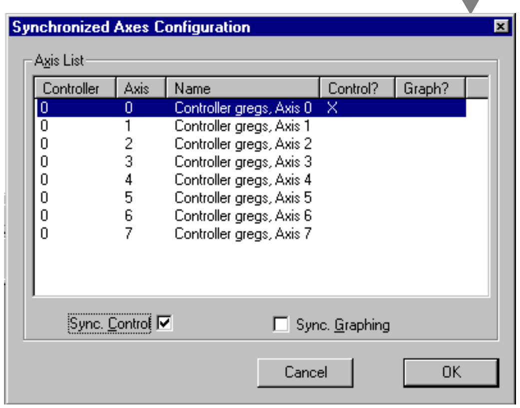
Hardware Summary



The *Hardware Summary* window is the first window you should access when setting up a new MEI controller. All controllers and their axes can be configured and the status viewed from the *Hardware Summary* window.

Sync Axes Toolbar

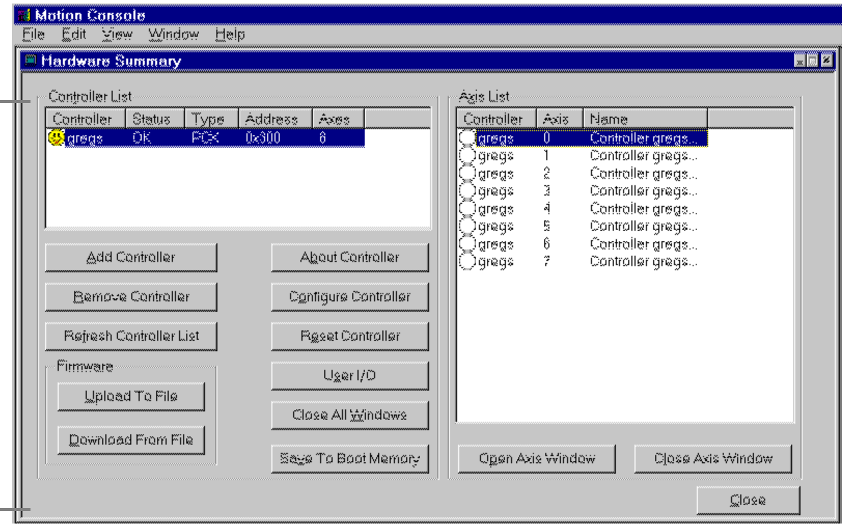




The *Synchronized Axes Configuration* window enables you to *quickly* configure axes for synchronous control and/or synchronous graphing.

Simply select an axis or axes, and then click in the desired box.

Hardware Summary



Controller List Group

Displays all currently installed and configured controllers in the system.
Use these functions to add, remove, and reset controllers, set and verify controller configurations, upload and download firmware, and configure User I/O.

All of the following buttons in the *Controller List* group operate on multiple controller selections with the exception of the buttons which open dialog boxes (*Add Controller*, *Configure Controller*, *Upload/ Download Firmware*). Buttons which are not applicable to the current selection will be disabled.

Controller List	Display the following information for all controllers in list: - controller name prededed by an icon indicating controller status - controller status: "OK" if addressable, or "Bad" if otherwise - controller type (e.g., LC, PCX, V6U, etc.) - controller address - number of axes of controller
Add Controller	Click on *Add Controller* to register a new controller with the system, including the controller *Name* and *Address*.
Remove Controller	Remove the selected controller(s) from the *Controller List*.
Refresh Controller List	Update the Configuration and Axis lists to reflect any status changes (i.e., axis disabled) that occured while Motion Console was running.
About Controller	Click on *About Controller* to display the controller type, firmware revision, and FPGA PROM version information that was obtained *from* the controller.

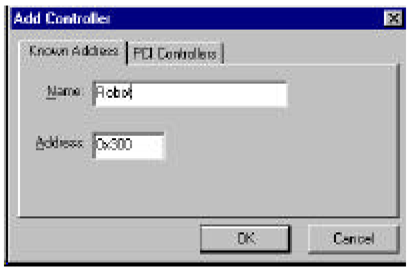
Configure Controller	Open the *Configure Controller* and select an axis (or axes) to enable/disable axes, set the controller's I/O address, or calibrate the DAC offsets.
Reset Controller	Reset controller(s) using the configuration and parameter settings stored in Boot Memory. (This is equivalent to a *dsp_reset(...)* function.)
User I/O	Open the *User I/O* window for the selected controller(s).
Close All Windows	Closes all windows associated with the selected controller(s).
Save to Boot Memory	Save firmware & configuration parameters to boot memory. MEI controllers include both volatile data memory and non-volatile boot memory, both of which you can access via Motion Console. Upon initialization, firmware and configuration parameters are loaded from boot memory to data memory, and then read by Motion Console. For example, when entering new tuning parameter values or configuration settings, Motion Console automatically stores these parameters in data memory. If the controller is powered off or reset, these data memory changes will be lost. Click the *Save to Boot Memory* button to save any changes to boot memory.

Hardware Summary

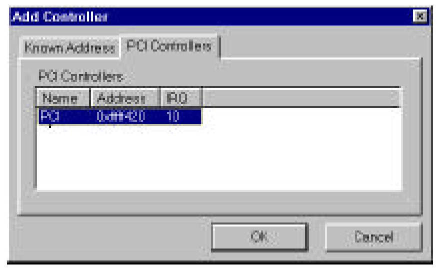
Warning! Only use Motion Console version 2.00.0006 or later with the PCI/DSP.

Add Controller

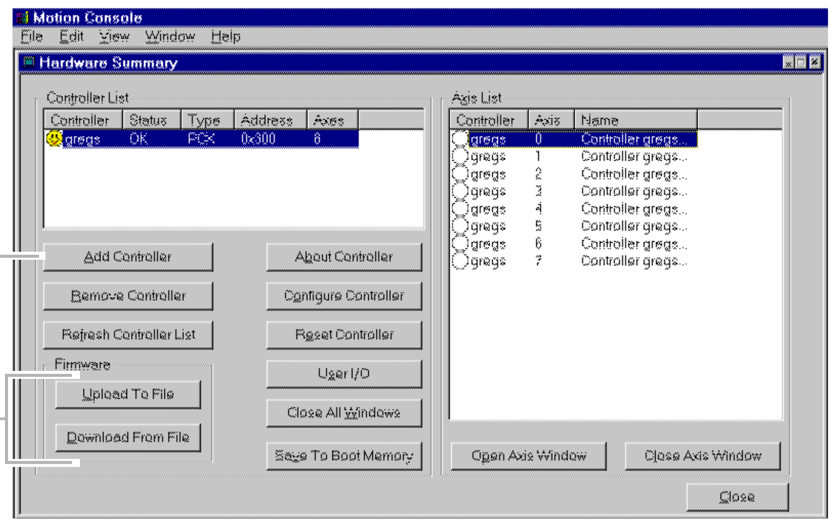
ISA Bus



PCI Bus



All PCI controllers will be listed here. Select the controller you want. If no controllers are found on the PCI Bus, no controllers will appear in this field.

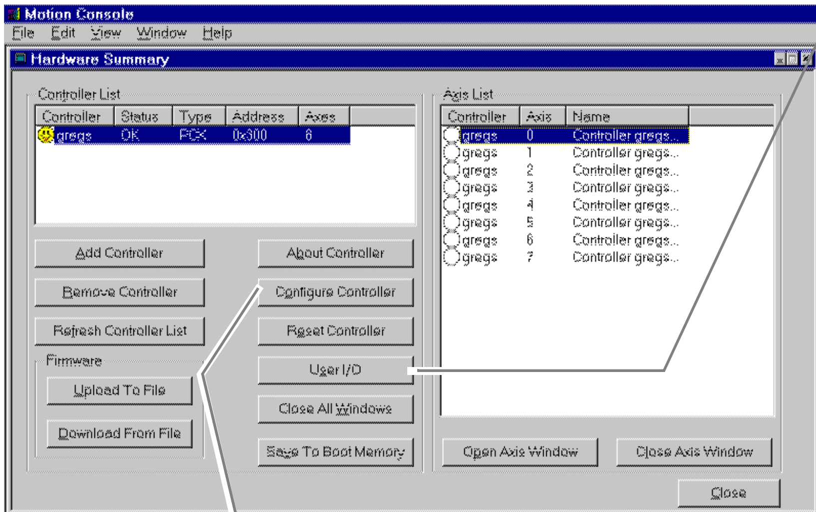


Firmware

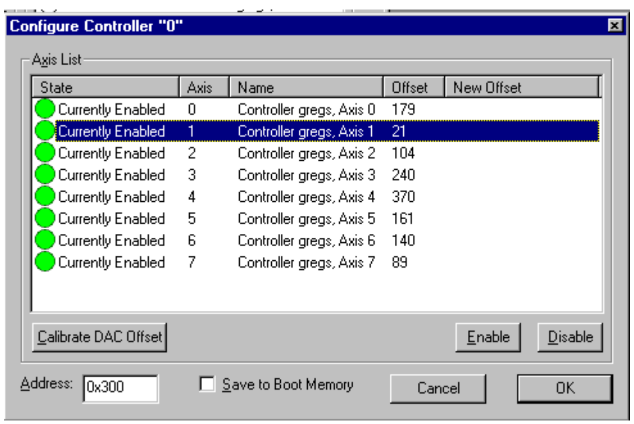
Upload to File	Copy controller's firmware to a file
Download from File	Download a "firmware" file to controller

Motion Engineering, Inc.

Hardware Summary

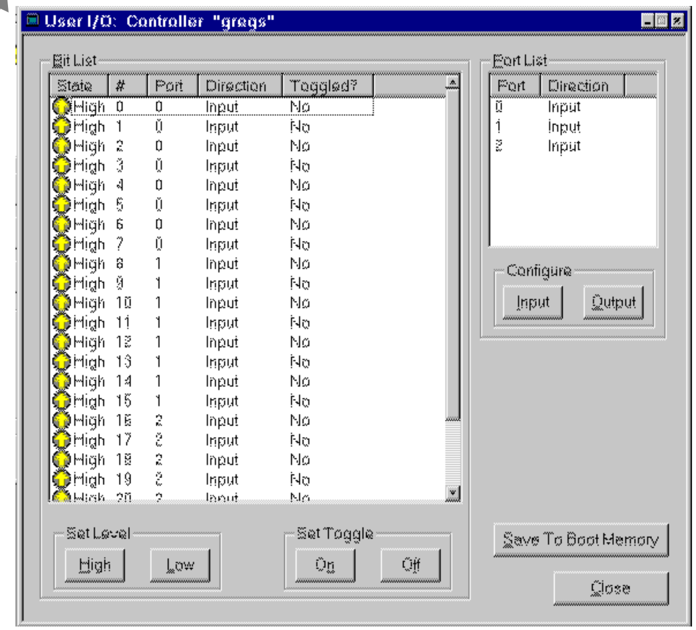


Configure Controller



The *Configure Controller* window enables you to enable/disable axes, set the controller's I/O address, or calibrate the internal DAC offsets. Simply select an axis or axes, and then configure as desired.

User I/O



Use this window to configure the motion controller's programmable User I/O lines.

Motion Console creates a *Port Group* box for each I/O port on the selected controller. MEI motion controllers can have from 3 to 6 port groups, each containing 6-8 lines (bits) depending on controller model. The *User I/O* window automatically displays the correct number and type of ports for the selected controller. Note that SERCOS controllers have no user I/O ports.

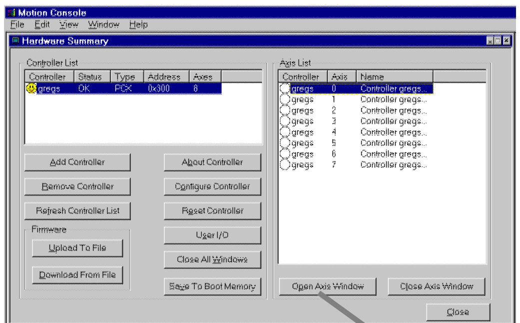
For each port group, the User I/O window provides the following sections and controls:

Configure (Input or Output)	Sets the group as *Input* or *Output*. *Input* groups can monitor for state changes on each I/O line indicated by *State* radio buttons. *State* radio buttons cannot be changed for input groups. *Output* groups can be set to a particular I/O state (using the *State* radio buttons) to test wiring and functionality. For *Output* groups, the *Toggle* box can also be checked to toggle output bit states.
State	For input groups, *State* indicates whether line is *High* or *Low*. For output groups, use *Set Level* buttons to configure individual lines as *High* or *Low*.
Set Toggle	Use in conjunction with Output mode. When selected, the bits on an output port will have their state toggled once per second.

Axis Window

Hardware Summary Window

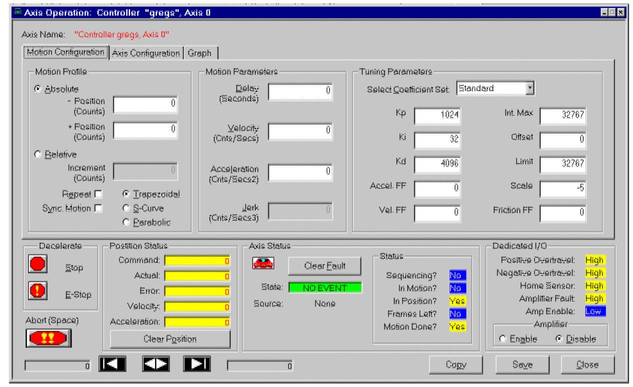
See pages B-3,4,5



Open Axis Window

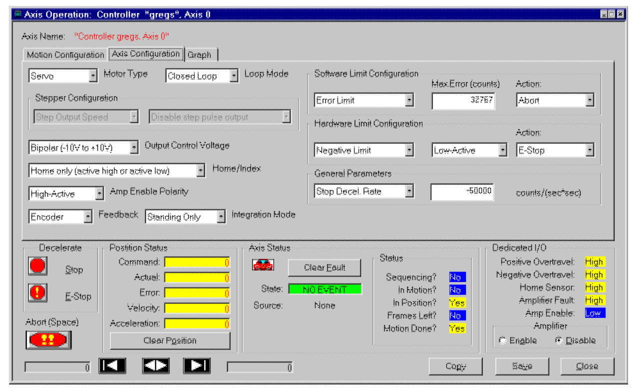
Use the *Open Axis Window (also called the Axis Operation Window)* to command motion, monitor status, and tune motors for the selected axis.

Before opening this window, select a **controller** from the *Controller List* and select an **axis** from the *Axis List*, then click *Open Axis Window*.



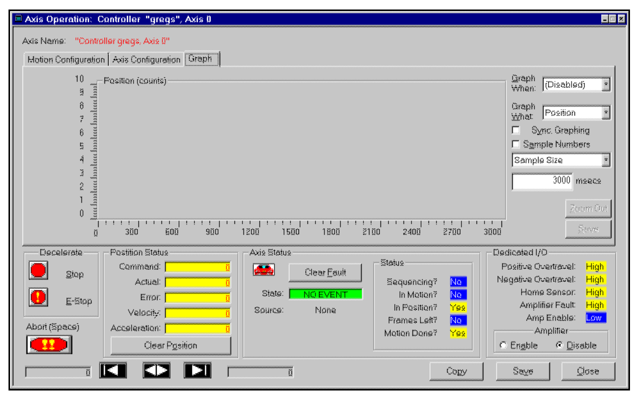
Motion Configuration Tab

See pages B- 8, 9



Axis Configuration Tab

See page B-10



Graph Tab

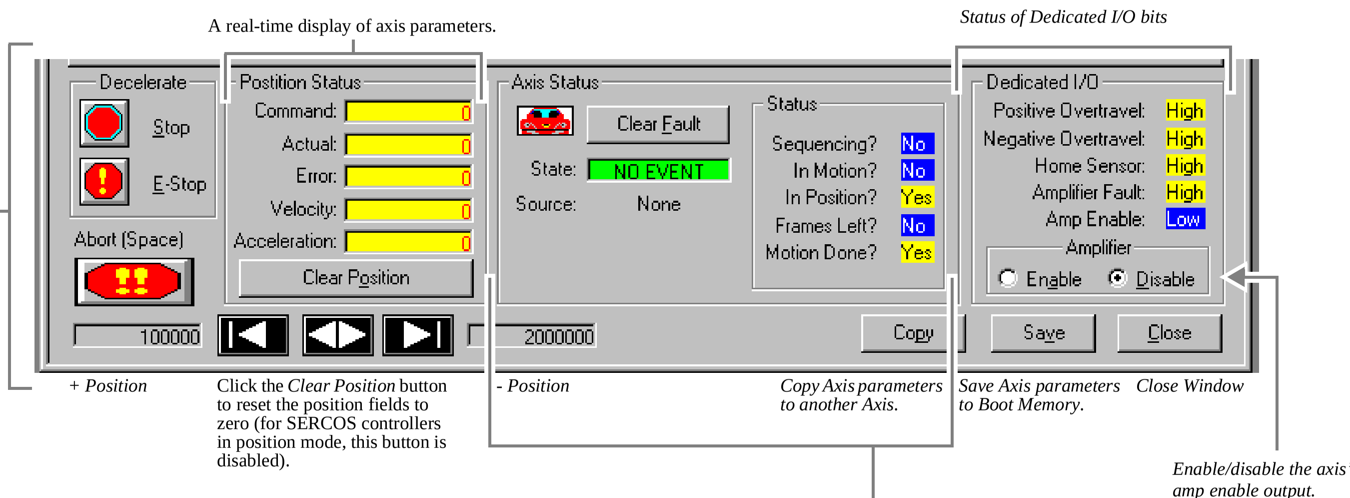
See page B-11

Axis Status/ Control Panel

See page B-7

Motion Engineering, Inc.

Axis Status/Control Panel



Movement Controls

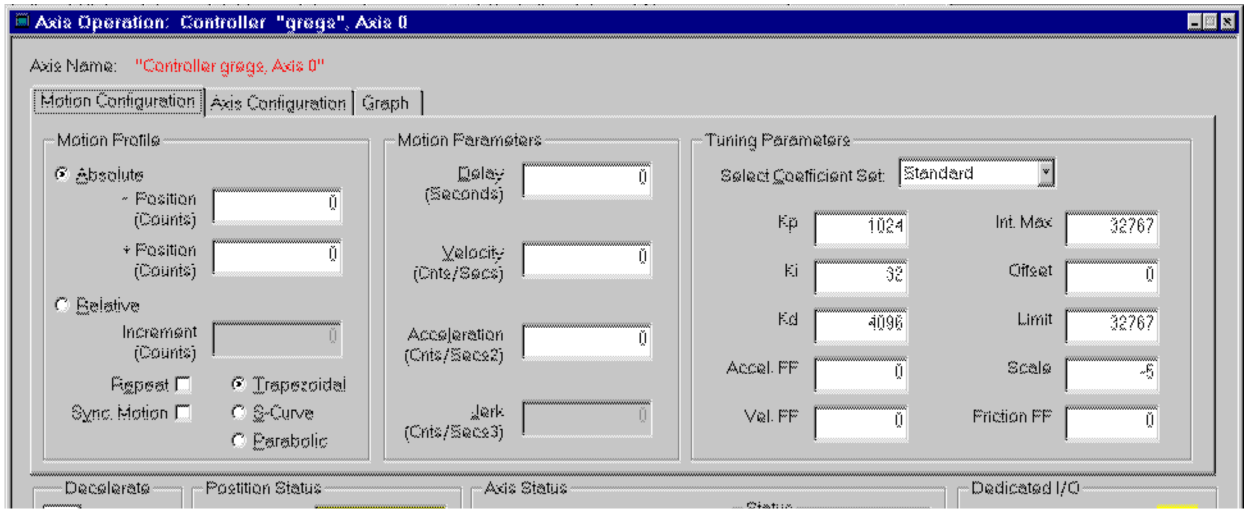
	In *Absolute Mode*: the ← button commands the motor to move to *- Position*, the → button commands the motor to move to *+ Position*. Clicking either button with *Repeat Mode* ON (in *Motion Profile*) starts a repetitve motion between *- Position* and *+ Position.* In *Relative Mode*: the ← button commands the motor to move to the current position **minus** the *Increment* value, the → button commands the motor to move to the current position **plus** the *Increment* value. Clicking either button with *Repeat Mode* ON (in *Motion Profile*) starts repetitive motion, in which the motor is continuously commanded to increment its position in the same direction.
	Clicking on the ← → button moves the axis to either + position or - position , whichever is farther from the current position.
Stop	Generates a stop event: immediately forces the axis to begin decelerating at the *Stop Deceleration* value (which is set on the *Axis Configuration* window).
E-Stop	Generates an emergency stop event: immediately forces the axis to begin decelerating at the *E-Stop Deceleration* value (which is set on the *Axis Configuration* window).
Abort (Space)	Immediately disables the *Amplifier Enable* and controller servo loop. (When the *Axis Window* is open, hitting your keyboard's <SPACE BAR> also generates an abort.)

Axis Status (Real-time display)

Yes (for logical True) or No (logical False)

Clear Fault	Reset any current faults on this Axis
State	Displays axis' current state (Running, No Event, Abort, Stop, E-Stop).
Source	Indicates source of any current axis faults
Sequencing?	Is the axis currently executing a motion sequence?
In Motion?	Is the axis currently in motion?
In Position?	Is the axis' *Actual Position* within the *In Position* window?
Frames Left?	Are there are any motion frames for the axis waiting to execute?
Motion Done?	Has all motion for the axis finished?

Motion Configuration



Motion Profile

Absolute Mode	Enables absolute motion as specified by values entered in *- Position* and *+ Position* fields. When *Absolute Mode* is selected, *Increment* is disabled.
- Postion	When *Absolute Mode* is selected, *- Position* specifies the position (in encoder counts) that the motor is commanded to move when the ← button is clicked.
+ Position	With *Absolute Mode* selected, *+ Position* specifies the position (in encoder counts) that the motor is commanded to move when the → button is clicked.
Relative Mode	Enables relative motion as specified by the value entered in the *Increment* field. When selected, *- Position* and *+ Position* are disabled.
Increment	When *Relative Mode* is selected, *Increment* specifies the number of encoder counts to reposition the motor when either the ← or the → buttons are clicked.

Work with Absolute and Relative Modes

Repeat	Specifies repetitive motion in both *Absolute* and *Relative Modes.* If *On*, axis starts repetitive motion that continues until *Repeat Off* or *Stop*, *E-Stop*, or *Abort* are clicked (or *Repeat Mode* is turned off).
Sync. Motion	Use when you want to synchronize the motion of 2 or more axes
Trapezoidal	Specifies a trapezoidal motion profile for current position
S-Curve	Specifies S-curve motion profile for current motion
Parabolic	Specifies a Parabolic motion profile for current motion

Motion Configuration Tab, cont.

Tuning Parameters Controls

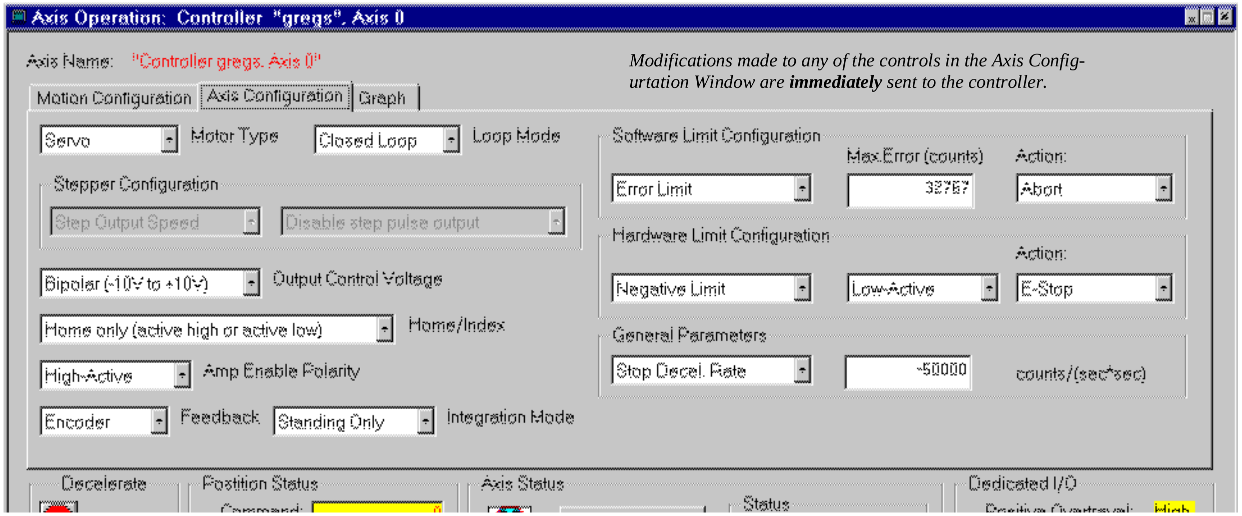
Use the *Tuning Parameters* controls to set an axis' control loop tuning parameters. The DSP-Series controllers use a second order PID algorithm with velocity and acceleration feed forward. Default parameters are shown in the figure.

Kp Proportional Gain	*Proportional Gain* determines the response of the system to position errors. Low Proportional Gain provides a stable system (doesn't oscillate), has low stiffness, and large position errors under load. Too large Proportional Gain values will cause oscillations and unstable systems. For velocity-controlled servos (voltage) and closed loop step systems, typical values are 100 - 500. For torque-controlled servos (current), typical values are 500 - 2000.
Ki Integral Gain	*Integral Gain* helps the control system overcome static position errors caused by friction or loading. The integrator increases the output value as a function of the position error summation over time. A low or zero value for the *Integral Gain* may have position errors at rest (that depend on the static or frictional loads and the *Proportional Gain)*. Increasing the *Integral Gain* can reduce these errors. If the *Integral Gain* is too large, the systems may "hunt" (oscillate at low frequency) about the desired position. Typical values are approximately 1/100th of the *Proportional Gain*.
Kd Derivative Gain	*Derivative Gain* provides damping by adjusting the output value as a function of *the rate of change of error*. A low value provides very little damping, which may cause overshoot after a step change in position. Large values have slower step response but may allow higher *Proportional Gain* to be used without oscillation. For velocity-controlled servos (voltage), typical values for *Derivative Gain* are roughly 2 times the *Proportional Gain* (200-1000). For torque-controlled servos (current), typical values are approximately 4 times the *Proportional Gain* (or 1000 - 8000).
Acceleration Feed Forward	*Acceleration Feed Forward* causes the controller to increase the output current during periods of acceleration and deceleration. Systems with high inertial loads need more motor current to accelerate or decelerate than systems with light loads need. *Acceleration Feed Forward* is used with torque-controlled servos (current).
Velocity Feed Forward	*Velocity Feed Forward* is useful in velocity-controlled servos or closed-loop stepper systems. As a speed of a system increases, the position error generally increases linearly and therefore a higher output voltage or pulse rate is required. The *Velocity Feed Forward* term reduces the *following error* by increasing the controller output voltage as a function of command velocity. If the *Velocity Feed Forward* is too large, the motor will try to travel ahead of the command position.
Integration Maximum	The integration of position errors is limited to a fixed DAC output value, the *Integration Maximum*. This prevents the integrator from "wind-up" in systems with high static friction. Set the *Integration Mode* (Active When Standing, or Active Always) in the *Axis Configuration* window.
Offset	The *Offset* term compensates for small variations in controller DAC outputs and amplifier offsets.
Limit	*Limit* prevents the 16-bit DAC output from exceeding a specified value. Typically, this value is reduced during initial tuning and set to full scale (32767), ±10V during normal operation, although some motor systems are designed to run at less than full scale values. For example, a 5V drive system would have a *Limit* of 16384 to prevent the output from exceeding 5V.
Scale	The *Scale* term enables the PID, *Vff*, and *Aff* terms to be scaled by the power of 2. Scale is limited to the range of -15 to 15. For example, a *Scale* value of 2 increases the filter terms by a factor of 4. A *Scale* value of -3 divides the the filter terms by a factor of 8.
Friction Feed Forward	The *Friction Feed Forward* term adds a constant value to the DAC output when the commanded velocity is non-zero. The sign of the value applied to the DAC is equal to *the sign of the command velocity* multiplied by the *Friction Feed Forward* term. The *Friction Feed Forward* term is 16-bits, and can range from -32767 to 32767. Torque-controlled motion systems with constant friction will benefit most from *Friction Feed Forward*.

Motion Engineering, Inc.

Axis Configuration Tab

Use to configure motor, feedback, home/index, and other parameters.



Motor Type

Servo
Stepper

Stepper Configuration

Disable step pulse output
Slow (0 to 20 kHz)
Medium (0 to 80 kHz)
Fast (0 to 325 kHz)
Superfast (0 to 550 kHz)

For servo motors, this parameter is *Disabled step pulse output.*

Output Control Voltage

Bipolar (-10V to +10V)
Unipolar (0 to +10V)

For step motors, the output control voltage should be set to *Unipolar*.

Hardware Limit Configuration

Negative Limit	Low-Active
Positive Limit	High-Active
Home	
Amp Fault	

For each parameter, configure active state and specify the *Action.*

Software Limit Configuration

Error Limit
Negative Limit
Positive Limit

For each limit, set *Max Error* and specify the *Action*.

Action

None
Stop
E-Stop
Abort

Loop Mode

Closed Loop
Open Loop

Stepper Mode

CW/CCW
Step/Dir

Amp Enable Polarity

Low-Active
High-Active

Integration Mode

Standing Only	Disables *Integral Gain* parameter when the command velocity is non-zero
Always	Enables the *Integral Gain* parameter **for all modes** of operation

Sets the PID integration mode for the axis.

General Parameters

Stop Decel Rate
E-Stop Decel Rate
Position Tolerance

For each parameter, specify counts/(sec*sec) or leave as default value.

Home/Index

Configures how the index pulse and/or the home input is used for homing
Home only (active high or active low)
Home Active **Low** AND High Index
Index only (active high or active low)
Home Active **High** AND High Index

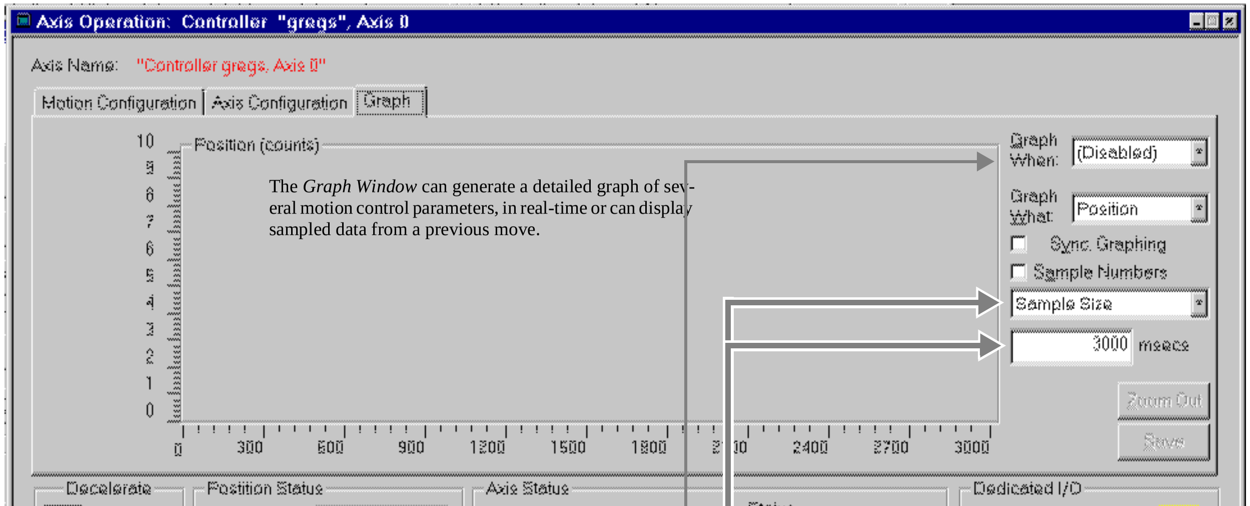
Feedback

Encoder	Supports single-ended and differential-ended incremental encoders
Parallel	Supports 32 discrete digital inputs converted to a 32-bit absolute position
Analog	Supports a 0 to 5V analog signal converted to a 12-bit digital position

Specifies the axis' actual position feedback source for PID algorithm.

B-10

Graph Tab



The *Graph Window* can generate a detailed graph of several motion control parameters, in real-time or can display sampled data from a previous move.

Graph When

(Disabled)	Disables updates to graphing display. Typically used to examine sampled data while continuing to perform moves.
Continuous	Displays a continuous, real-time graph of the commanded motion. New data is shown *to the left of* the moving cursor line; old data is shown *to the right of* the cursor line. Entering a new sample size while graphing discards the current data. It is possible to zoom in on graphs generated in *Continuous* mode.
On Move	Starts graphing at the start of the next move after the Refresh button is clicked.
On Command	Starts graphing when the Command button is clicked.

Graph What

Position	Actual *and* Command Position
Error	Position Error (in counts)
Velocity	Actual *and* Command Velocity
Voltage	Actual voltage of the axis' servo output

Sync. Graphing	To graph more than 2 or more axes at a time
Sample Numbers	Changes horizontal scale to parameter selected in 2 boxes below it

Use these 2 boxes to dynamically change these parameters during continuous graphing. In the first box (upper), you select the parameter to be changed, while in the lower text box, you enter the desired value.

Sample Size	Jerk	Ki	Offset
Sample Size *Cont.*	Position 1	Kd	Limit
Delay	Position 2	Accel. FF	Scale
Velocity	Increment	Velocity FF	Friction FF
Acceleration	Kp	R.I. Max	Offset

APPENDIX C SETUP.EXE

Intro

For DOS, Win 3.x & Win 95/98 Only

The SETUP program is a powerful tool for installation, configuration, tuning and debugging for PC-based architectures running DOS, Windows 3.x, and Windows 95/98. We recommend that you use Motion Console for Windows, Windows 95/98, and Windows NT systems.

Note that SETUP.EXE will not work as a "DOS" window under Windows NT.

SETUP's main screen has pull-down menus that are used to access different windows. Many windows can be accessed and arranged on the screen at one time. Each window will enable you to see and manipulate the command position, actual position, dedicated I/O, software limits, axis status, axis state, source of an event, etc. for each axis.

Important!	**Before you write any code**, we recommend that you 1. Use the SETUP program to thoroughly test the hardware 2. Make sure that you can perform two-point motion (using repeat) with **all** of your motors If you do not have motors connected to the controller, you can simulate the motors by configuring the axes as open-loop steps (unipolar).

To Load the SETUP Program

The "Setup" CD-ROM contains the SETUP program, the firmware (.ABS files) and the CONFIG program.

1. On your hard drive (C: or whatever), create the directory C:\MEI\SETUP and copy the files from the "Setup" CD to that directory.

2. Next run the SETUP program by typing SETUP at the DOS prompt. You should next see the *About SETUP* window, which shows the date and version of the SETUP program.

   Note that when SETUP initializes the controller, SETUP does not change *any* of the current configurations or conditions on the DSP Series controller.

Mouse/Trackball

A mouse or trackball makes the SETUP program much easier to use.

Hot Keys

If you do not have a mouse or trackball, you can use the keyboard to perform the same tasks.

Table E-1 Hot Keys

Hot Key	
Space Bar or <ENTER>	Select the highlighted button
F2	Open a **Position Status** window
F3	Open an **Axis Status** window
F4	Display Motion Graphics
F5	Move the current window (with the cursor keys)
F6	Jump to the next open window
F7	Open **Tuning Parameters** window
F8	Open **Axis Configuration** window

Table E-1 Hot Keys

Hot Key	
F9	DSP hardware reset
<ESC>	Close the current window
Alt/F	Select the **File** menu
Alt/C	Select the **Configure** menu
Alt/S	Select the **Status** menu
Alt/M	Select the **Motion** menu
Alt/X	Exit the SETUP program
Cursor Keys	Move between fields and buttons

Buttons

In each window, there are buttons provided to send, read and save information stored in the controller's **data memory** (volatile) and **boot memory** (non-volatile).

Table C-2 Buttons

Button	
Send	**Write** the **values** in the window **to data memory**. (Same as the <ENTER> key).
Set Axis	Set the axis to **display** the current **values in data memory**. (Same as the <ENTER> key).
Save	**Store** the window **values to boot memory**.
Read	**Copy values** from boot memory **to data memory**.
Copy All	**Copy** the **values** in the window **to data memory** for all axes. Values displayed in other windows are not affected.
Save All	**Store** the **values** in the window **to boot memory** for all axes. Values displayed in other windows are not affected.

Saving Default Parameters to the Controller

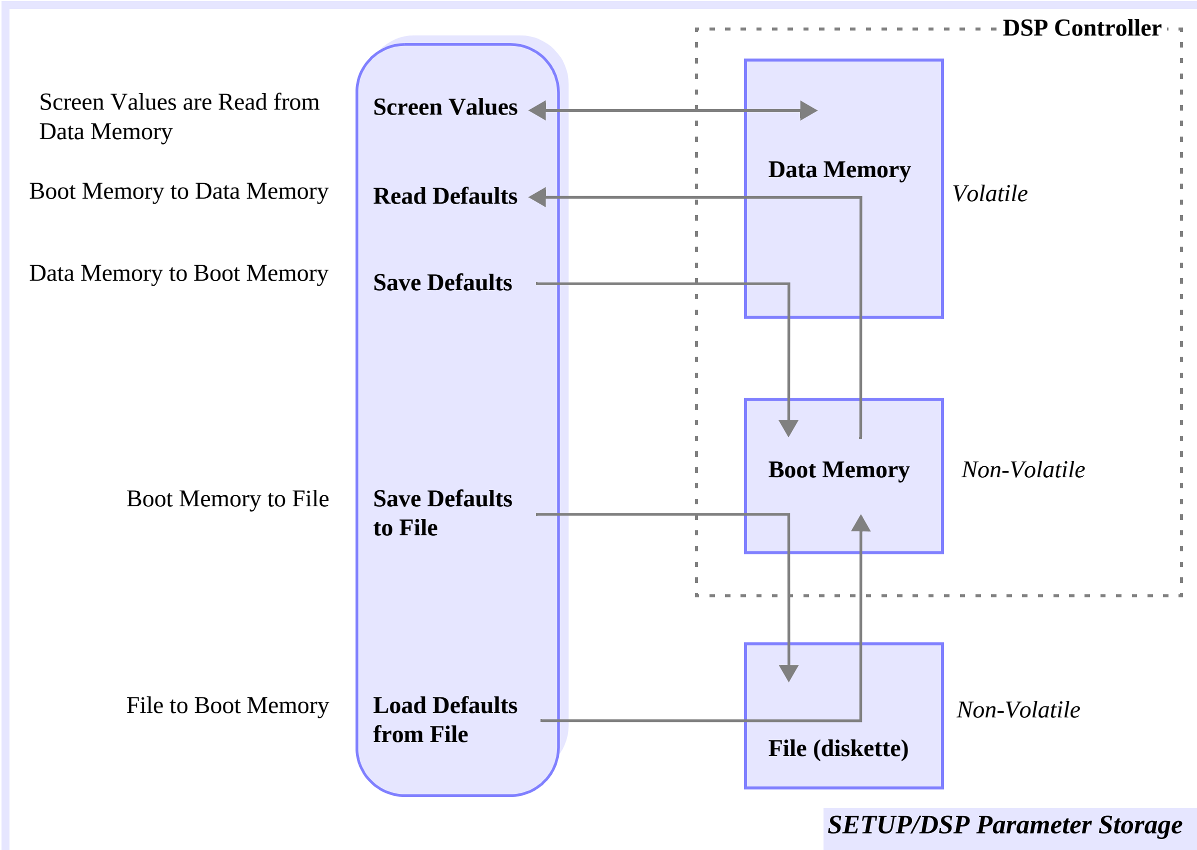
Many of the configuration windows have *Read* and *Save* buttons. The *Read* button loads the default configuration parameters (power-up or reset) from boot memory to data memory. The *Save* button stores the current parameters to boot memory.

The SETUP program can access the data memory (volatile) and boot memory (non-volatile). When a value is entered in any window, the value is *automatically stored* in the controller’s data memory. The values stored in data memory are lost when the controller is reset (F9 key) or when the power is turned off.

The reset function (F9) loads the firmware and configuration parameters from boot memory to data memory. During initialization, the SETUP program reads the values stored in boot memory.

Save Defaults to File saves the current boot memory configuration to a diskette file with the extension *.ABS*. *Load Defaults From Disk* loads the boot memory configuration from a diskette file into boot memory.

Figure C-1 SETUP's Default Parameters Storage



Functional Grouping by Axis

Some of the functions and parameters of the controller must be the same across groups of axes:

Table C-3 Functional Grouping by Axis

Function	*Number of Axes*	*Window Selections*
Step or Servo Motor	2	If a 3-axis controller is to be used for 2 stepmotors and 1 servo motor, then the servo motor must be axis 2 and the step motors axes 0 and 1. When a pair of axes (2 and 3 in this case, even though axis 3 is not present) are configured as a servo axes, the step pulse output is turned off for both axes 2 and 3.
Open-loop or Closed-loop	2	If a 3-axis controller is to be used for 2 closed-loop step motors and 1 open-loop step motor, then the open-loop step motor must be axis 2 and the closed-loop motors must be axes 0 and 1.
Home & Index Functions	4	On a 4-axis controller, all axes will be configured to use the Home and Index in the same fashion. On a 7-axis controller, axes 0-3 will have a configuration that is independant of the configuration for axes 4-6.

SETUP Menus & Screens

The SETUP menus and screens are organized under 4 main categories:

Table C-4

Menu	*Windows*	*Parameters in that window*	*Page*
File	Load default parameters from a diskette to file Store default parameters to a diskette file Shell out to DOS Display version number Move location of selected window Exit program		*page C-6*
Configure	Set I/O address Set PID tuning parameters Set auxiliary tuning parameters Set axis configuration Set limit switch configuration Set software limits Reset controller with boot memory	 Servo or step Open/closed-loop Stepping speed Home sensor configuration Voltage output Feedback device type Integration active mode	*page C-8*
Status	Monitor position status Monitor axis status Monitor dedicated I/O status	Position, Velocity, Acceleration and Error Idle/Run Mode, In Motion, In-Position, Source and State Enable/Disable Amplifier	*page C-15*
Motion	Two-point motion Graphic motion analysis	Endpoints, Delay, Velocity, Acceleration and Jerk Motion Profile: Trapezoidal, Parabolic, or S-Curve	*page C-17*

File Menu

The File menu contains these options:

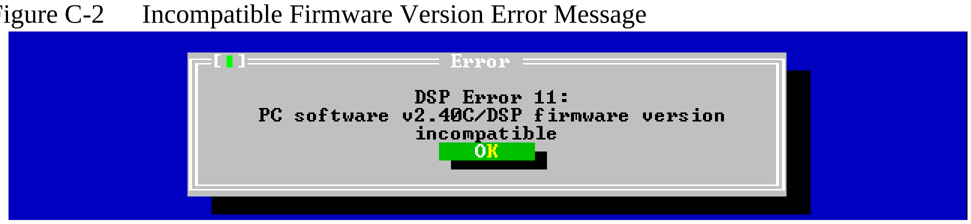
Option	
Load Defaults from File	Loads the values from CD into DSP boot memory (requests filename)
Save Defaults to File	Saves values in DSP boot memory to diskette (prompts for filename)
DOS Shell	Shells to DOS
About	Displays the program version number
Exit	Terminates the program

Load Defaults from File

Selecting *Load Defaults from File* will read a CD file containing the firmware, which includes the parameters for the PID filter, limit switch configurations, software limit configurations, etc. After the prompt appears, you can select the desired filename.

Tip! Incompatible Firmware Version	If SETUP displays a message stating that the version of SETUP is *incompatible* with the firmware currently installed on the controller, then: 1. Exit SETUP (Alt + X) 2. Turn off the amplifiers and/or drivers 3. Run the CONFIG.EXE program (found on the SETUP disk) 4. Run the SETUP program The CONFIG program will download new firmware to the static RAM on the controller, and set the internal offsets to zero the output. Note that previously stored tuning parameters, etc., will be erased.

Figure C-2 Incompatible Firmware Version Error Message



Save Defaults to File

Selecting *Save Defaults to File* will write the firmware, PID parameters, limit switch configurations, software limit configurations, etc., to a diskette file. After the prompt appears, you can select the desired filename. We recommend that after configuring the controller, you store your configuration into a file on a diskette. Once stored on diskette, the parameters can be easily downloaded to the board in the future if necessary.

DOS Shell

Selecting *DOS Shell* allows you to access the DOS command line without exiting the SETUP program. Type EXIT at the DOS prompt to return to the SETUP program.

About

Selecting *About* displays the SETUP version number and date.

Exit

Exits the SETUP program. After exiting, motion will stop, but all configuration parameters will remain active.

Configure Menu

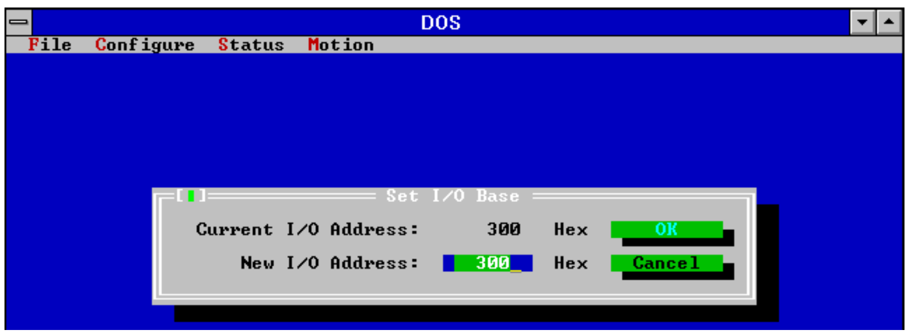
The Configure menu contains the options

Option/Window	
I/O Base Address	Sets the I/O address where SETUP comminicates with the controller
Tuning Parameters	Sets tuning parameters, DC offset and voltage/pulse rate limit
Aux. Tuning Parameters	Sets auxilliary tuning parameters: derivative sample rate, etc.
Axis Configuration	Allows axes to be configured as step/servo, etc.
Limit Switch Configuration	Sets the active level of limit switches and associated action
Software Limits	Sets the software limits and associated actions
Reset	Resets the DSP with parameters stored in battery backed RAM

I/O Base Address

Use the *I/O Base Address* window to set the base address for the controller. If SW1 is set for an address other than 300 hex, you must use the *I/O Base Address* window to tell the SETUP program the location of the controller, or use the DSP environment variable (in DOS) to set the address of the controller.

Figure C-3   Configure/Set I/O Base Address Window



Using the DSP "Base" variable to set the controller's address

The SETUP program also has the ability to read an environment variable called 'DSP' and automatically set the base address. Currently, SETUP only understoods the "BASE" parameter of the DSP variable, which you can use to specify the base I/O address of the controller. If you specify the BASE parameter, then the SETUP program will initialize the controller using the 'BASE' address. For example, if '`set DSP=base: 0x280`' is executed at the DOS prompt, then the SETUP program will use address 280 hex. (The CONFIG program will also use this address).

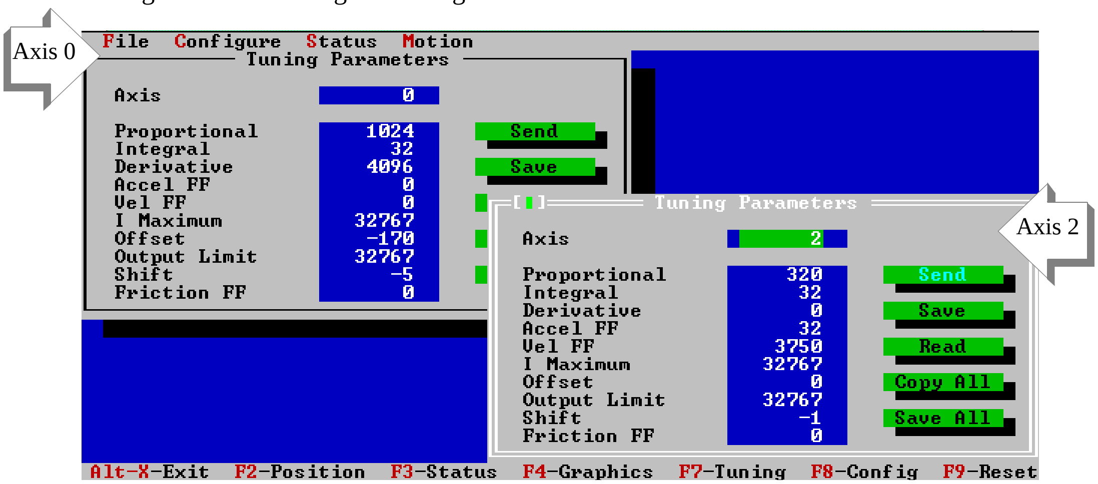
Tip! DSP Not Found	If SETUP displays a message that the DSP controller cannot be found at the specified address, be sure that the DIP switches on the controller are set for the same address entered on the CONFIGURE/SET I/O BASE window. If SETUP still displays a message that the *DSP is Not Found*, press the F9 key to re-execute the SETUP program. If the SETUP program still cannot find the DSP, run the CONFIG program.

Tuning Parameters

Use the *Tuning Parameters* window to set the control loop tuning parameters for each axis. The DSP uses a second order PID algorithm with velocity and acceleration feed-forward.

Note that multiple types of windows can be open simultaneously. For example, windows can be open for *Tuning Parameters* and *Motion Status* for one axis, or windows can be open for *Tuning Parameters* and *Motion Status* for both Axes 1 and 4. This makes it possible to change tuning parameters on-the-fly and to observe the effect in real time.

Figure C-4 Configure/Tuning Parameter Windows for Axes 0 and 1



The PID algorithm is based on the following formula:

$$On = Z^{shift} (Kp*En + Kd*(E_n - E_{n-1}) + Ki * Sn + Kv*Vn + 64*Ka*An) + Kf * Mn + Ko$$

The lower case *n* represents the sample period. The terms are defined as follows:

if -Smax < Sn < Smax	then	S*n* = S*n*-1 + Ei
if Sn > Smax	then	S*n* = Smax
if Sn < -Smax	then	S*n* = Smax

On = DAC output
Kp = proportional gain
Ki = integral gain
Ka = acceleration feed-forward
Kf = friction feed-forward
Mn = 0 or 1 based on the command velocity
An = command acceleration * 2-6
Smax = maximum integrated error

$\mathbf{Z^{shift}}$ = overall scale factor
Kd = derivative gain
Kv = velocity feed-forward
Ko = static DAC offset
En = position error
Vn = command velocity
Sn = integrated error

Proportional Gain

The proportional gain affects the analog command voltage or pulse rate based on the amount of position error. The higher the proportional gain, the "stiffer" the response.

If the proportional gain is set **too low**, the response will be "mushy" - the motor will have trouble following the commanded trajectory.

If the proportional gain is set **too high**, the motor may oscillate or "buzz" at rest or during motion. The range of values for proportional gain is 0 to 32,767.

Integral Gain

Use the integral gain parameter to integrate static errors and "fine tune" the position at rest. The motor command (analog voltage or pulse rate) will increase with increasing error and time. The maximum amount of gain due to integration is limited to prevent "windup."

With proper tuning, motor sizing and a low-friction mechanical system, 0-1 encoder count (step) error is possible. The range of values for integral gain is 0 to 32,767.

Derivative Gain Use the derivative gain term like a damping factor. The derivative gain affects the analog command voltage or pulse rate based on the amount of position error change occurring in the last two samples. The range of values for the derivative gain is 0 to 32,767.

Acceleration Feed-Forward
Use the acceleration feed-forward term to add extra output during acceleration to reduce following error. The range of values for the acceleration feed-forward is 0 to 32,767.

Velocity Feed-Forward
Use the velocity feed-forward term to add extra output during constant velocity to reduce following error. The range of values for the velocity feed-forward is 0 to 32,767.

I Maximum Use the *I Maximum* limit to prevent "windup." Generally, "windup" occurs in systems where (very) high friction cannot be overcome without entering an oscillation mode. The *I Maximum* parameter sets the maximum voltage output by the integration term of the PID algorithm. The range of values for the integration limit is 0 to 32,767.

Offset Use the *Offset* parameter to compensate for other system offsets. The *Offset* parameter sets the DAC output level. However, in most cases, the offset parameter should be set at 0.

Note that each axis also has an *internal offset*, which is in series with the *digital filter offset* (*Offset* parameter, which is visible on the SETUP program tuning screen). You use the *Offset* parameter to zero the DAC and Voltage-to-Frequency converter outputs, to prevent motion when the axis is placed in idle mode.

The internal offset is set by the CONFIG.EXE program. After the CONFIG program has run, the normal DAC offset should be under 3 millivolts (which will not produce step pulses). Temperature drift is approximately 1 millivolt per degree C.

Only positive values of *Offset* will output steps, since the Voltage-to-Frequency converter can only react to positive voltages. The range of values for the *Offset* is +/- 32,767.

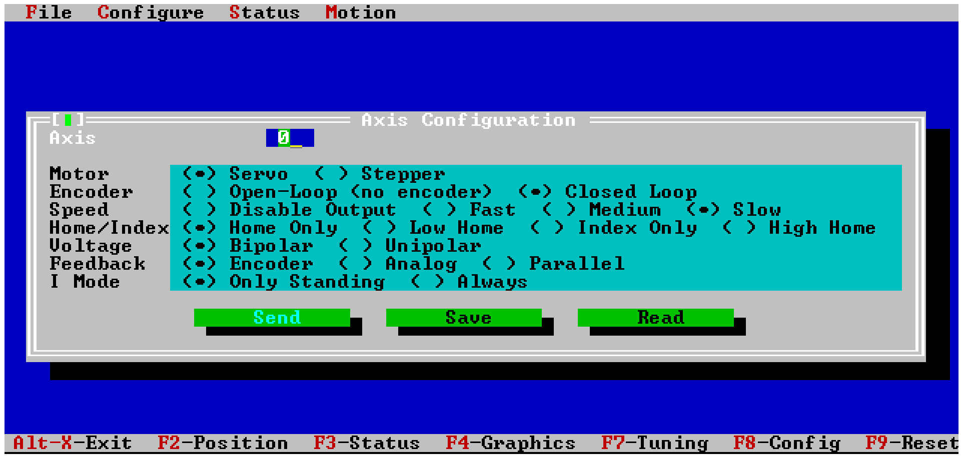
Output Limit Use the *Output Limit* parameter to limit the controller output (analog voltage or pulse rate) during system tuning. For servo motors, this term limits the analog output voltage. For step motors, this term limits the step pulse output rate. The range for the output limit is 0 to 32,767. This range corresponds to -10V to +10V for servos (i.e. 0.000305 volts/unit) or 0 to full scale pulse rate for steps.

Shift Range Use the *Shift Range* parameter to shift the range of the tuning parameters. The shift factor multiplies or divides all the filter parameters by a user specified power of two. For example, if the shift factor = -3, then all the filter parameters will be divided by 8 (2^{-3} = 1/8). If the shift factor = 2, then all the parameters will be multiplied by 4. This is useful for unusual motors such as air-bearing motors, voice-coil actuators and hydraulics or other actuators. The default parameter is -5, i.e. a multiplier of 2^{-5} or 1/32.

Friction Feed-Forward
Use the friction feed-forward term to add extra output during any commanded velocity, to reduce *following error* caused by friction. The range of values for the friction feed-forward are 0 to 32,767.

Axis Configuration

Figure C-5 The Configure/Axis Configuration Window



Motor

Servo *or* **Stepper**
This selection is used to enable/disable the step pulse output for a given pair of axes. Selecting "step" will enable the step output for the pair of axes (0 and 1, 2 and 3, etc.). The analog output is available regardless of the selection. When the motor type is changed, a set of default tuning parameters will be loaded into SETUP for that axis.

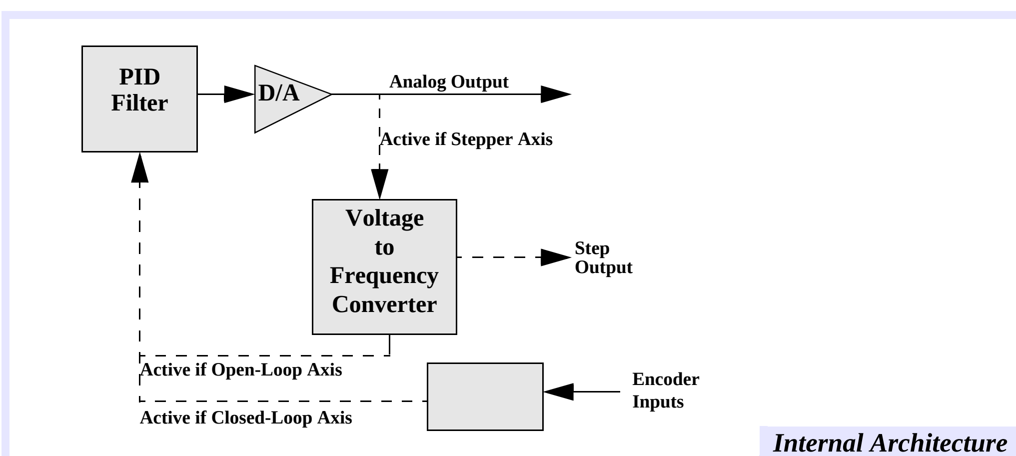
Encoder

Open-loop *or* **Closed-loop**
This selection enables you to indicate if a pair of axes is to be open-loop or closed-loop.

If closed-loop is selected, the controller will use feedback from an external device to close the loop.

If open-loop are selected (and **Step** was selected on **Motor** line), the board will direct the step output back into the encoder input for the axis, in effect digitally closing the loop on the controller

Figure C-6 Internal Architecture to Control Step Motors



Speed

Disable Output *or* **Fast** *or* **Medium or Slow**
This selection sets the maximum pulse rate for the step output in either open-loop or closed-loop mode. Whenever **Step** is selected (on **Motor** line), the step speed range must be set. The

ranges are:

Slow	*Medium*	*Fast*
0 to 23 kHz	0 to 94 kHz	0 to 375 kHz

You must set the tuning parameters as follows for each axis configured for open-loop steps:

Table C-5 Tuning Parameters for Open-loop steps

Parameter	*Setting*
Proportional (K_p)	320
Integral (K_i)	32
Derivative (K_d)	0
Accel FF (K_a)	32
Vel FF (K_v)	3750
I Maximum	32767
Offset	0
Output Limit	32767
Shift-	-1 (Slow), -3 (Medium), -5 (Fast)

We recommend choosing the slowest possible speed range that is adequate for your system. If an axis is configured for servo, the speed selection should be: **Disable Output**.

Home/Index

Home Only *or* **Low Home** *or* **Index Only** *or* **High Home**
This selection configures whether the index pulse is required for the home input to be active. Typically a rotary encoder has a single index pulse (per revolution). The index pulse can be used with the home signal input to produce accurate homing to within one encoder count. Standard boards have four possible types of homing.

Type	*Description*
Home Only	Home input only (active high or active low)
Low Home and Index	Home input ANDed with index (active low home and active high index)
Index Only	Index only (active high or active low)
High Home and Index	Home input ANDed with index (active high home and active high index)

Note that the home/index setting affects the axes in groups of 4. For example, on a 4-axis controller, all the axes must be configured the same with respect to home/index. For more information, see the section on home switch wiring.

Voltage

Bipolar *or* **Unipolar**
This selection configures whether the analog output is unipolar (0 to +10V) or bipolar (-10V to +10V).

If you are using analog servo motors, configure the output for bipolar operation.
If you are using stepper motors, configure the output for unipolar operation.

Tip!	
If the Motor Turns Only in 1 Direction	**Stepper motors:** If a step motor turns in only one direction, check the *Configuration/Axis Configuration* window to be sure the axis is set for UNIPOLAR. The voltage-to-frequency-convertor only responds to positive voltages (unipolar) and will not output steps if the voltage is negative (bipolar). **Servo motors:** If a servo motor turns only in one direction, check the *Configuration/Axis Configuration* window to be sure the axis is set for BIPOLAR.

Feedback

Encoder *or* **Analog** *or* **Parallel**
This selection allows each axis to be configured for the type of feedback device used. The choices are:

Selection	*Device Type*	*Pin Location*
Encoder	Incremental Encoder	Motor Signal Header
Analog	Unipolar LVDT	Analog Input Header
Parallel	Laser Interferometer	User I/O Headers

Note that each axis can be individually configured with any feedback device, but if any axis uses analog inputs, the remaining analog inputs cannot be used for any purpose other than analog feedback. For example, the analog inputs on the same controller cannot be used for both a joystick *and* analog feedback.

I Mode

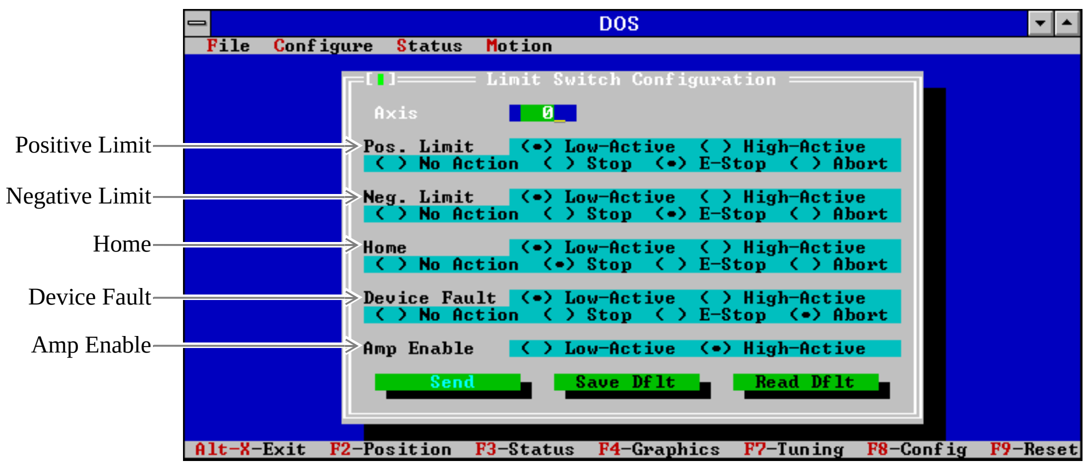
Only Standing *or* **Always**

The I Mode (integration mode) selection allows the PID integration term for each axis to be configured as:

Only Standing	Only when the command velocity is zero
Always	During motion and when standing

Limit Switch Configuration

Figure C-7 Configure/Limit Switch Configuration Window

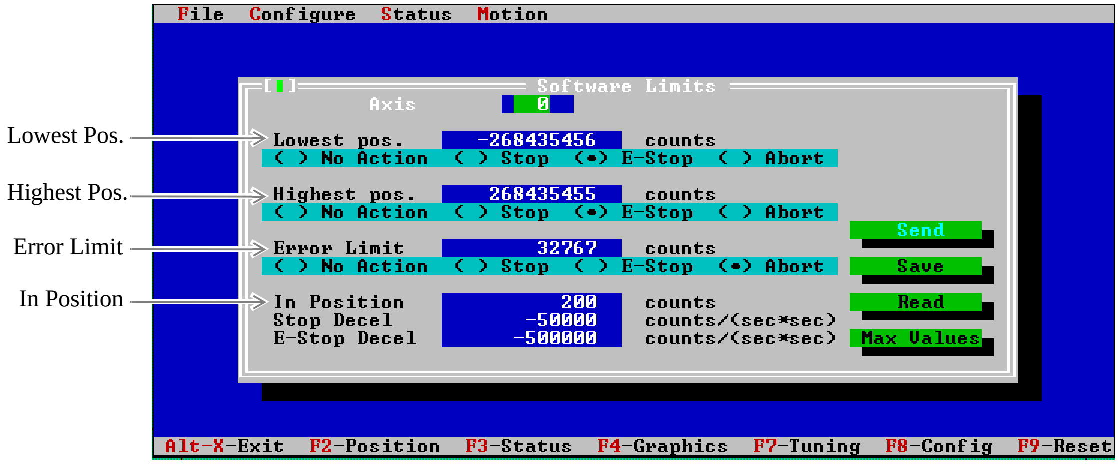


The *Configure/Limit Switch Configuration* window defines the active state of the positive/negative limit switches, home switch, device fault and the amp enable output. It also specifies which event is triggered when each sensor becomes active. The events are:

Event	
No_Action	Ignore a condition
Stop	Decelerate to a stop (at specified stop rate)
E-Stop	Decelerate to a stop (at specified E-stop rate)
Abort	Disable PID control and the amplifier for this axis

Software Limits

Figure C-8 Configure/Software Limit Configuration Window



This window is used to set the software limits (lowest position, highest position and error limit) for each axis. The values for each of these limits, and the event to be performed when the limit is exceeded, can be specified.

Event	
No Action	Ignore a condition
Stop	Decelerate to a stop (at a specified stop rate)
E-Stop	Decelerate to a stop (at a specified E-stop rate)
Abort	Disable PID control and the amplifier for this axis

Reset (F9)

This selection will perform a power-up reset of the DSP controller. The software and hardware configurations are re-read from boot memory, the command and actual positions are reset, the amp enable output is disabled, User I/O is reconfigured, etc. A hardware reset causes the DSP to release control of the axes and I/O for a few milliseconds which may cause motors to jump.

Status Menu

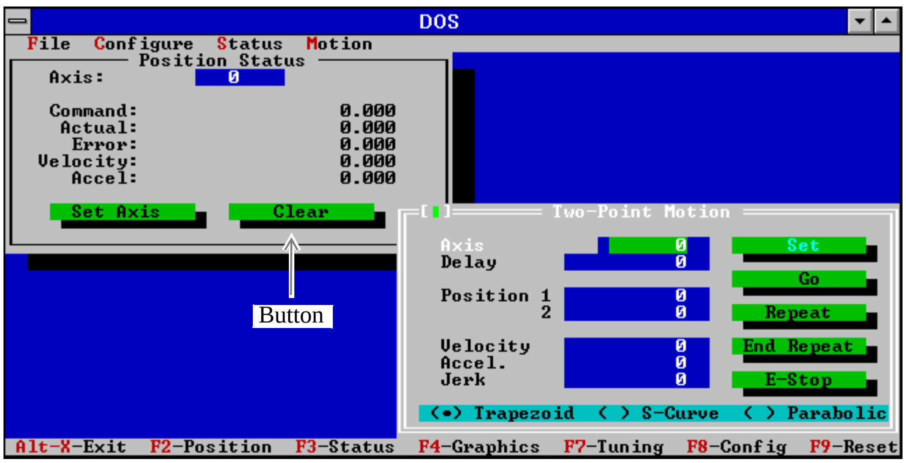
The Status menu contains the options:

Option/Window	
Position Status	Displays the position, velocity and acceleration of each axis
Axis Status	Displays the status of each axis: Motion, E-stop, Run/Idle, etc.
Dedicated I/O	Display the status of dedicated I/O lines

Position Status

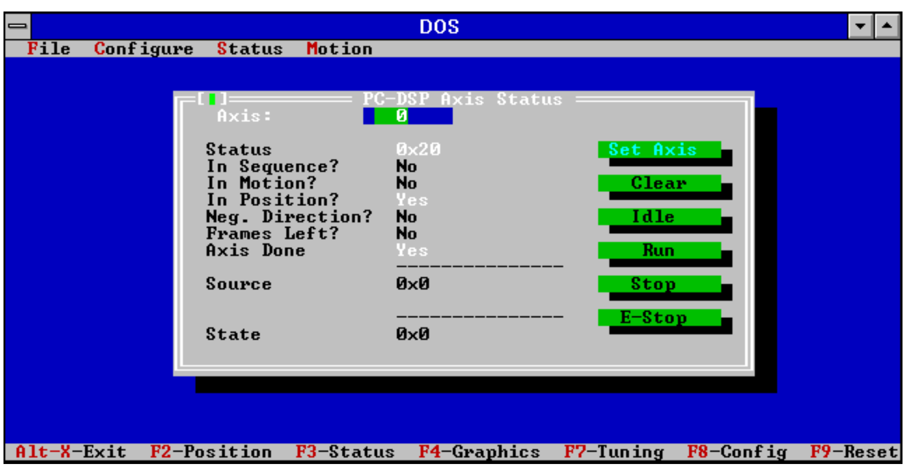
The *Position Status* window is a read-only window which provides an easy way to monitor the status of each axis. The *Clear* button will immediately zero actual and command positions.

Figure C-9 Status/Position Status Window



Axis Status

Figure C-10 Status/Axis Status Window



This window displays the real-time status of the flags for the axis displayed. In the following description the term “Event” means Stop, E-Stop, or Abort. The status items reported are:

Item	
Status	Displays the current condition of an axis in hex.
In Sequence?	Displays “Yes” if a set of frames describing a move is executing.
In Motion?	Displays “Yes” if command velocity is non-zero.
In Position?	Displays “Yes” if the position is within the in-position window.
Negative Direction?	Displays “Yes” if the command velocity is negative.
Frames Left?	Displays “Yes” if additional, unexecuted frames are still in buffer.
Axis Done?	Displays “Yes” if *In Motion?* is “No” and *In Position?* is “Yes.”
Source	Displays the source (host CPU, position limit, etc.) of a current event.
State	Displays current event on an axis.

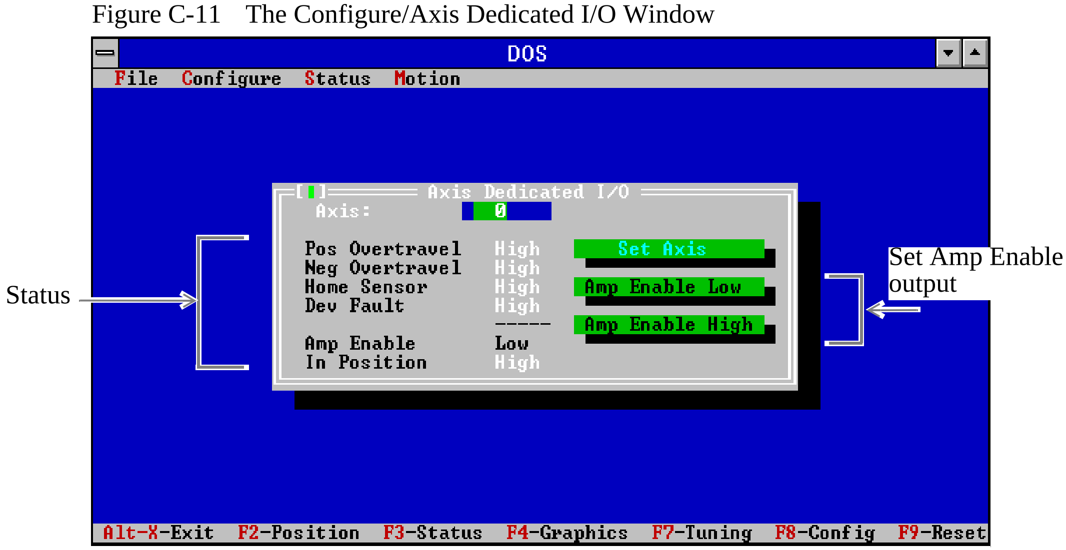
The buttons on the right of the window perform the following functions:

Button	
Clear	Reset all flags, clear stops and E-stops.
Idle	Set analog and step/direction outputs to zero, disables amp enable output and disables PID filter.
Run	Closes the loop, enables PID filter. Note that loop is closed internally (on-board) for open-loop steps when Run mode is selected. Amp enable must be manually enabled.
Stop	Decelerate at Stop rate.
E-stop	Decelerate at E-Stop rate.

Dedicated I/O

The *Dedicated I/O* window displays the status of the dedicated inputs for limits, home, device fault and in-position. The window also contains buttons to set the *amp enable* output to a high or low state.

Figure C-11 The Configure/Axis Dedicated I/O Window



Motion Menu

The Motion menu contains the options:

Option/Window	
Point-to-Point Motion	Commands an axis to move between two points
Graphics Analysis	Displays the command vs. actual and analog outputs for a move

Point-to-Point Motion

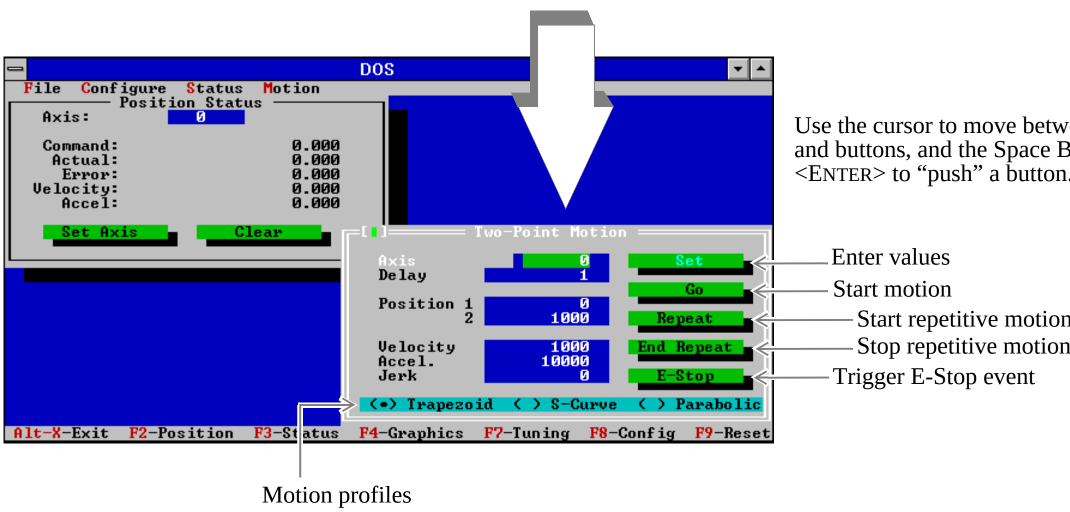
Use the *Point-to-Point Motion* window to command motion between two points. *Point 1* and *Point 2* specify the endpoints of the motion, *Velocity* specifies the maximum slew speed and *Acceleration* the acceleration rate. The *Jerk* field is only used when performing non-constant acceleration profiles (S-curve and parabolic). Units are encoder counts (steps), counts (steps) per second, counts (steps) per second^2 and counts (steps) per second^3.

Use the GO button to start the motion. Use the *Repeat* and *End Repeat* fields to start or stop repetitive motion. Use the E-STOP field to trigger an E-Stop event. Use the cursor to move between fields and buttons, and the space bar or <ENTER> to "push" a button.

Three motion profiles are available: trapezoidal, parabolic and S-curve. **Generally, choose an acceleration that is 10 times the velocity and a jerk that is 100 times the acceleration.**

Recall that the velocity is the *rate of change* of the position, acceleration is the *rate of change* of the velocity, and jerk is the *rate of change* of the acceleration. Also note that increasing the velocity and acceleration of parabolic and S-curve moves can actually increase the time to position.

Figure C-12 The Motion/Two-Point Motion Window



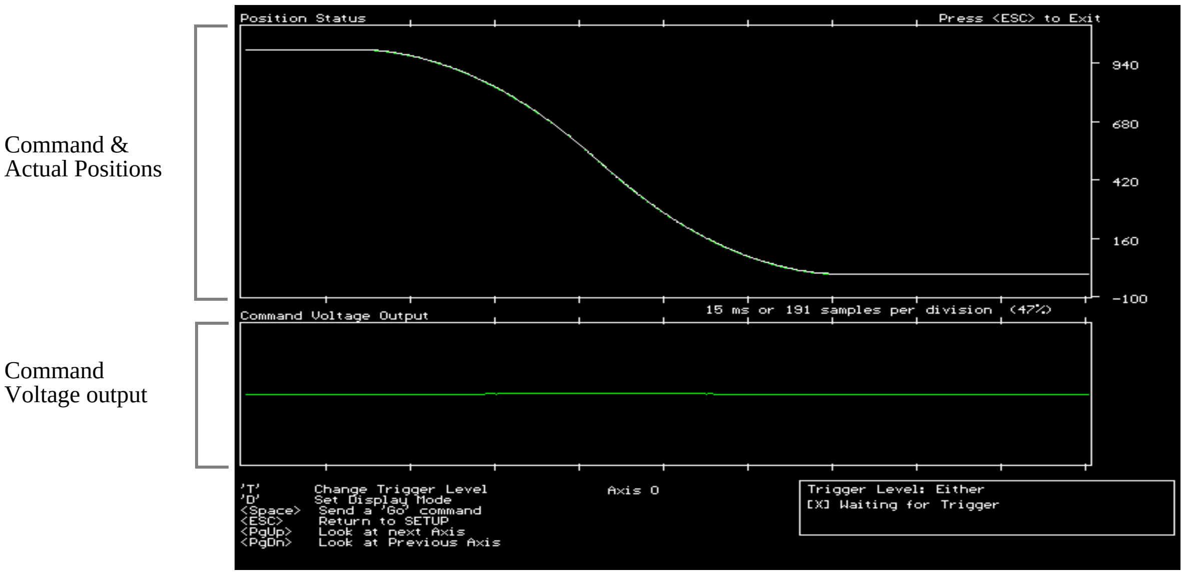
The *Delay* field allows motion to be paused at the endpoints. Units of delay are *relative time* and depend on the computer's CPU speed.

Graphic Analysis

The *Graphics Analysis* window provides a visual guide to tuning closed-loop systems. Motion is controlled by the parameters set on the *Motion/Two-Point* window. Trapezoidal, parabolic or S-curve motion may be commanded. Endpoint positions, velocity, acceleration and jerk may also be selected.

Command and actual position are overplotted on the graphic screen. A second plot shows the analog voltage output on the same time scale.

Figure C-13   Sample Graphic Analysis Screen



You can access the graphics screen directly from any window by pressing the F4 key. To display continuous motion, use the “Repeat” button in the *Motion/Two-Point Motion* window. To command single-step motion, press the space bar. The hot keys that control data acquisition and display are:

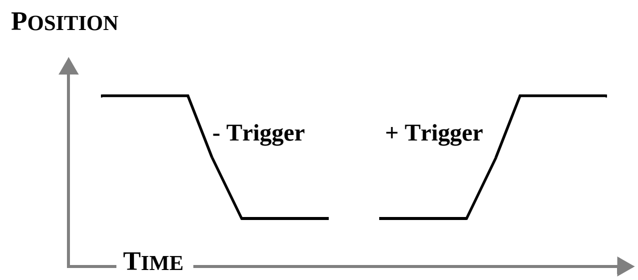
Table C-6   Hot Keys

Key	
<Space Bar>	Command a single move - acquire data in either direction
T	Change the trigger mode for REPEAT motion
+	Trigger data acquisition on increasing position
-	Trigger data acquisition on decreasing position
D	Set Display mode
C	Continuous sample and display
S	Collect data then display
PGUP	Change to next higher axis number
PGDN	Change to next lower axis number
Arrow Keys	Change the time scale during motion (left and right keys)

Trigger Mode

When the graphic screen is displayed, press the “T” key to display the motion *in one direction only*, triggering on rising or falling position counts (steps). Note that when a rising or falling trigger mode is used, motion *in one direction only* (every other move) will be displayed.

Figure C-14 Trigger Modes



Trigger Modes

Display Mode

Press the letter	*to select*
D	Display mode
C	Continuous mode, which instructs the host CPU to collect and display the data simultaneously
S	Sample-then-display mode, which instructs the host CPU to collect all the data, and then display it on the screen. The sample-then-display mode has a higher sampling rate (since the host is not printing data) and will provide more accurate data for very fast moves.

Note that the screen includes a percentage value representing the number of data points displayed divided by the total number of DSP calculation cycles during the move.

APPENDIX D TUNING

Intro

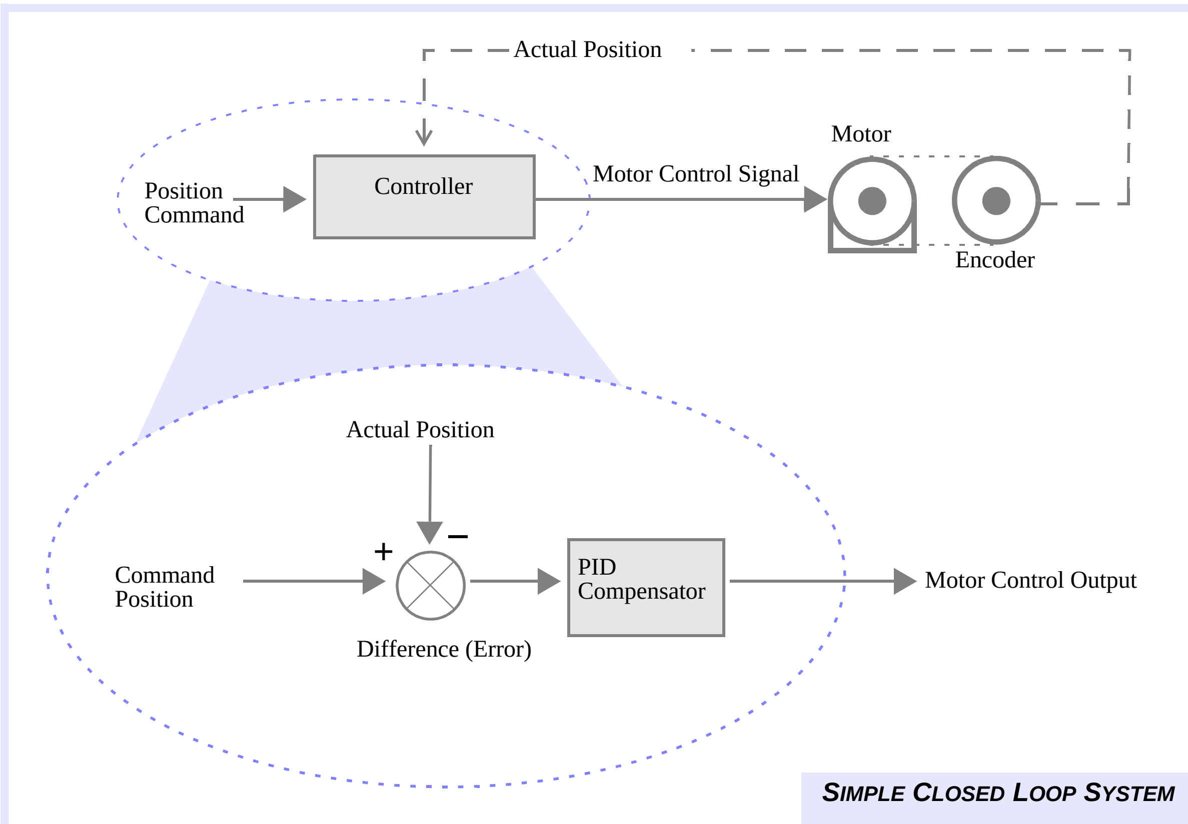
In closed-loop positioning systems, the motion controller compares the command position (trajectory) to the actual position feedback and calculates a motor control signal. The position error is defined as the difference between the command and actual positions. As the position error increases, the motor control signal (analog output or step pulse rate) is increased to counteract the error. The digital filter coefficients (PID, Proportional Integral Derivative) determine the computation of the value of the motor control signal based on the position error.

Tuning is the process of adjusting these digital coefficients to provide the best control for a particular system of motors and loads.

There are 2 methods generally used for tuning closed-loop digital control systems: calculation or trial-and-error. Calculation involves rather complex mathematics and precise knowledge of all of the system parameters such as motor and amplifier response, load inertia and friction. Control systems textbooks provide methods for calculation of the tuning parameters for a large variety of applications.

Trial-and-error has the advantage in that no knowledge of the control system's possesive parameters is necessary and no calculations are needed. However, you may need to try a large number of trial parameters to tune a system and some combinations of parameters may produce an unstable or runaway system. An *organized approach* to searching for the best combination of tuning parameters helps shorten the tuning time while avoiding an unstable combination which may damage the system.

Figure D-1 Simple Closed-Loop System



The Digital Filter

The DSP calculates an axis' output (analog voltage or pulse rate) based on a PID servo control algorithm. The current position error is the input to the PID algorithm. The current position error is the difference between the command position and the actual position. The actual position is controlled by the feedback device, and the command position is determined by the trajectory calculator. The trajectory is based on the commanded motion profiles from software. The PID algorithm is based on the following formula:

$$On = 2^{shift} (Kp*En + Kd*(E_n - E_{n-1}) + Ki*Sn + Kv*Vn + 64*Ka*An) + Kf * Mn + Ko$$

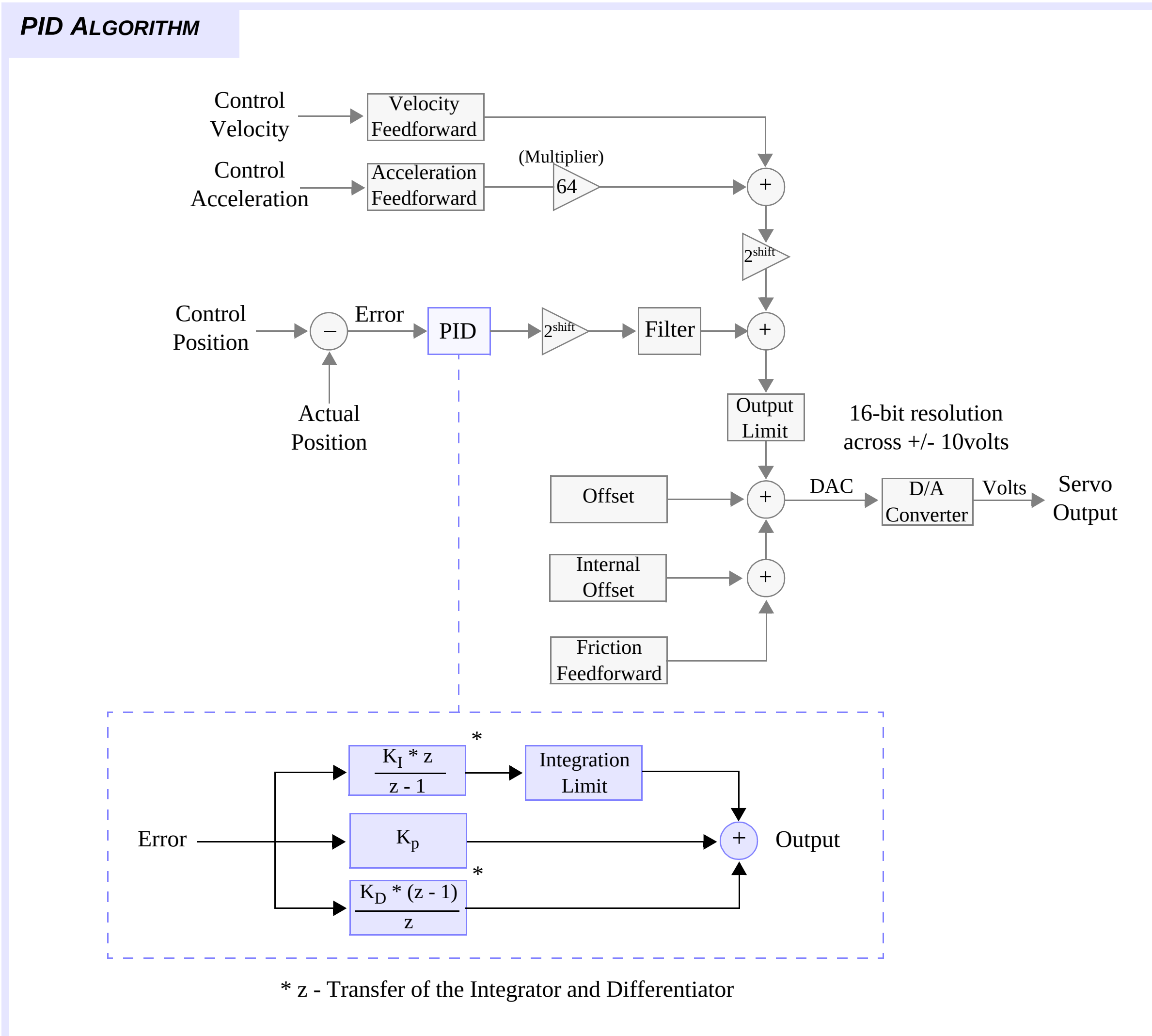
The lower case "n" represents the *sample period*. The terms are defined:

if -Smax < Sn < Smax	Sn = Sn-1 + En
if Sn > Smax	Smax
if Sn < -Smax	-Smax

On = DAC output
Kp = proportional gain
Ki = integral gain
Ka = acceleration feed-forward
Kf = friction feed-forward
Mn = 0 or 1, based on the command velocity
An = command acceleration
Smax = maximum integrated error

^{shift} = overall scale factor
Kd = derivative gain
Kv = velocity feed-forward
Ko = static DAC offset
En = position error
Vn = command velocity
Sn = integrated error

Figure D-2 PID Algorithm



Tuning Parameters

Table D-1 What do Gains Do?

	Parameter	
Kp	Proportional Gain	increases/decreases the motor control output based on the **position error** of the current sample
Kd	Derivative Gain	increases/decreases the motor control output based on the **rate of change of the position error**
Ki	Integral Gain	increases/decreases the motor control output based on the **summation of position error** over time
Kv	Velocity Feed-Forward	increases/decreases the motor control output based on the **command velocity**
Ka	Acceleration Feed-Forward	increases/decreases the motor control output based on the **command acceleration rate**
Kf	Friction Feed-Forward	Adds a **constant value** to the motor control output when the command velocity is non-zero.
Ko	Offset (static)	Adds a **constant value** to the motor control output
2^{shift}	Scale	**Scale factor** for the other tuning parameters (Kp, Kd, Ki, Kv, Ka, Ko)
	Integration Limit	Limits the **summation of position error** over time.

Table D-2 What Problems Do Gains Solve?

	Parameter	
Kp	Proportional Gain	Determines the systems' overall response to **position error**
Kd	Derivative Gain	Provides damping and stability for the system by **preventing overshoot**
Ki	Integral Gain	Helps the system overcome **static position errors** (caused by friction or loading)
Kv	Velocity Feed-Forward	Increases the system's motor control signal based on the command velocity (useful for amplifiers in **velocity mode**).
Ka	Acceleration Feed-Forward	Increases the system's motor control signal (current) during acceleration and deceleration (useful for amplifiers in **torque mode**)
Kf	Friction Feed-Forward	Increases the system's motor control signal (current) during acceleration and deceleration to overcome static friction (useful for systems with **large dynamic friction loads**)
Ko	Offset (static)	Compensates for **small variations** in motor control signal due to DAC and amplifier offsets (also used to compensate for a fixed force, like gravity
2^{shift}	Scale	Adjusts the resolution of the gains and feed-forward terms (via a **scale factor**)
	Integration Limit	Prevents the integrator from building up a large integration error (and consequently **saturating** the motor control signal)

Proportional Gain (Kp)

The Proportional Gain determines the overall response of a system to position errors. The Proportional Gain increases/decreases the motor control output signal based on the position error.

Table D-3 Effects of Proportional Gain

If Proportional Gain is	***System tends to be***	***with Stiffness***	***and incurs under Load***
Low	very stable and to not oscillate	low	large *position errors*
High	less stable and oscillate	high	small *position errors*

Table D-4 Typical Proportional Gain Values

For	***Typical Proportional Gain Values are***
Velocity-controlled servos (voltage) or closed-loop step systems	**100-500**
Torque-controlled servos (current)	**500-2000**

When Proportional Gain is Too Low

The motor (Actual Position) is unable to keep up with the command position if the Kp term is too small. At the beginning of the move, the motor falls behind and the voltage output is slow to respond. Eventually, the voltage will reach a level that can compensate for the error. Then, as the position error decreases, the voltage will also begin to decrease. This decrease in voltage will again cause the motor to fall behind. The end result is that the output voltage and position error will oscillate, as demonstrated in the graphs below.

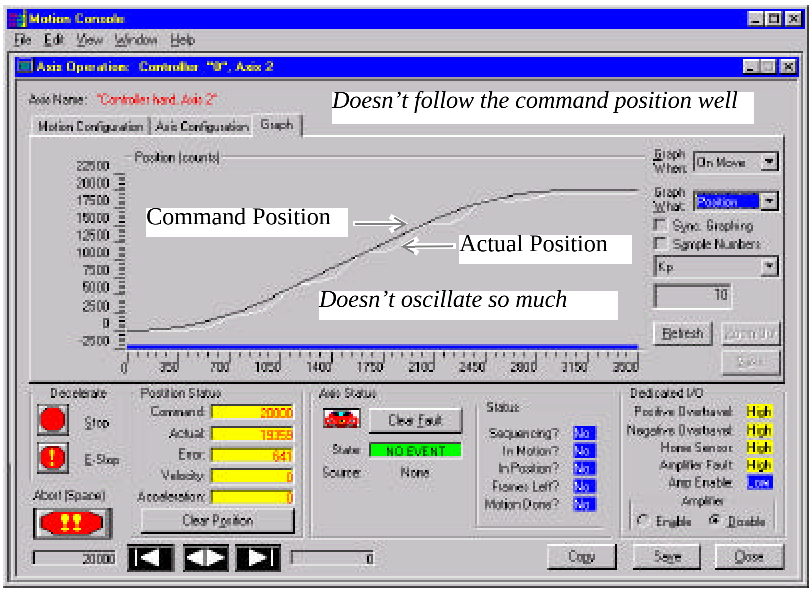
Also, low Kp values will often result in static errors at the end of moves.

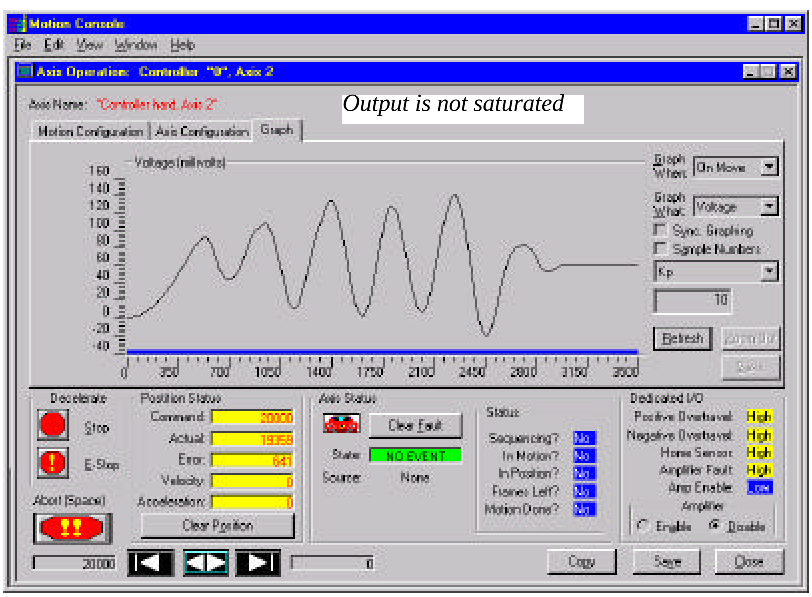
Figure D-3 Insufficient Proportional Gain

LOW PROPORTIONAL GAIN

The motor (actual position) is not keeping up with the command position, yet the output voltage is not saturated. Note the static error at the end of the move.

More gain is needed.





When Proportional Gain is Too High

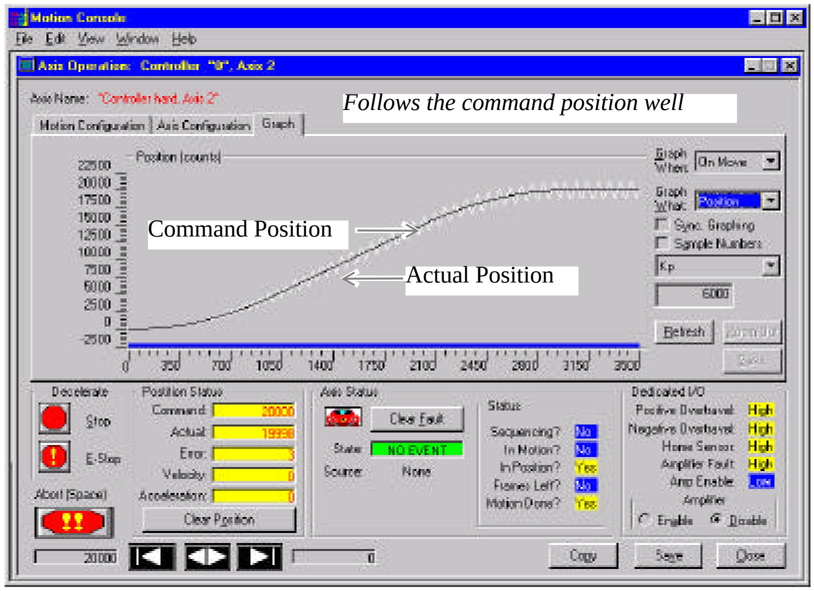
The motor (Actual Position) is able to keep up with the Command Position, but the motor oscillates and the voltage saturates. Due to the high gain, the output responds very strongly to any position error. As a result, the output signal saturates.

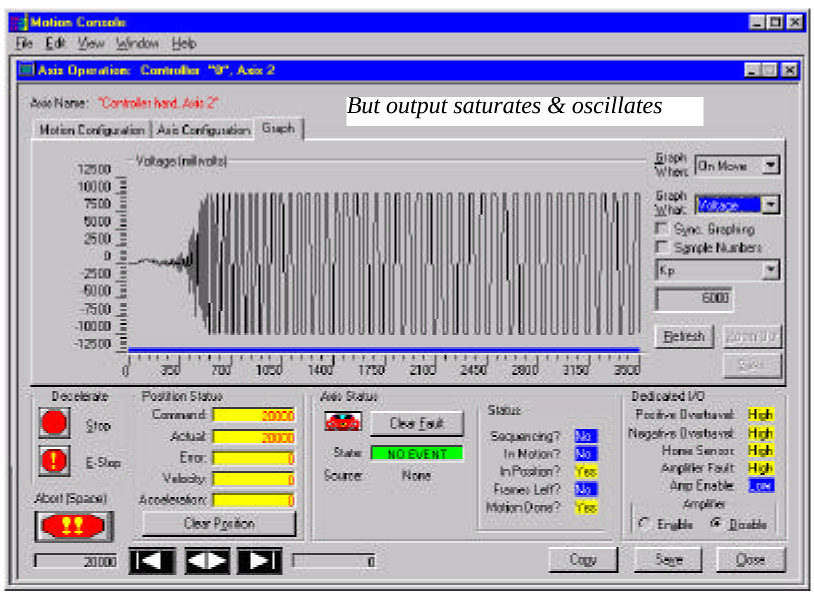
Figure D-4 Excessive Proportional Gain

HIGH PROPORTIONAL GAIN

The motor (actual position) is keeping up with the command position, but the output voltage is saturated and oscillates.

Less gain is needed.





Note:	Excessive proportional gain is characterized by oscillation. In some situations, damping (derivative gain) can be increased to help compensate.

Derivative Gain (Kd)

The Derivative Gain increases/decreases the motor control output signal, based on the *rate of change of the position error*. The Derivative Gain provides damping and stability to the system, by preventing overshoot.

Table D-5 Effects of Derivative Gain

If Derivative Gain is	*System Response*
Low	very fast, but has overshoot (ringing)
High	not as fast but may allow higher Proportional Gains to be used (without oscillation)

A low value for the Derivative Gain causes the system to have a very fast response to changes in position error, but also to have a possible overshoot or "ringing" after a step change in position. Large values of Derivative gain have a slower step response, but also may allow higher Proportional Gain to be used *without oscillation*.

Table D-6 Typical Derivative Gain Values

For	*Typical Derivative Gain Values are*
Velocity-controlled servos (voltage)	**200-1000**, roughly *2 times the Proportional Gain*
Torque-controlled servos (current)	**1000-8000**, roughly *4 times the Proportional Gain*

Figure D-5 Insufficient Derivative Gain

LOW DERIVATIVE GAIN

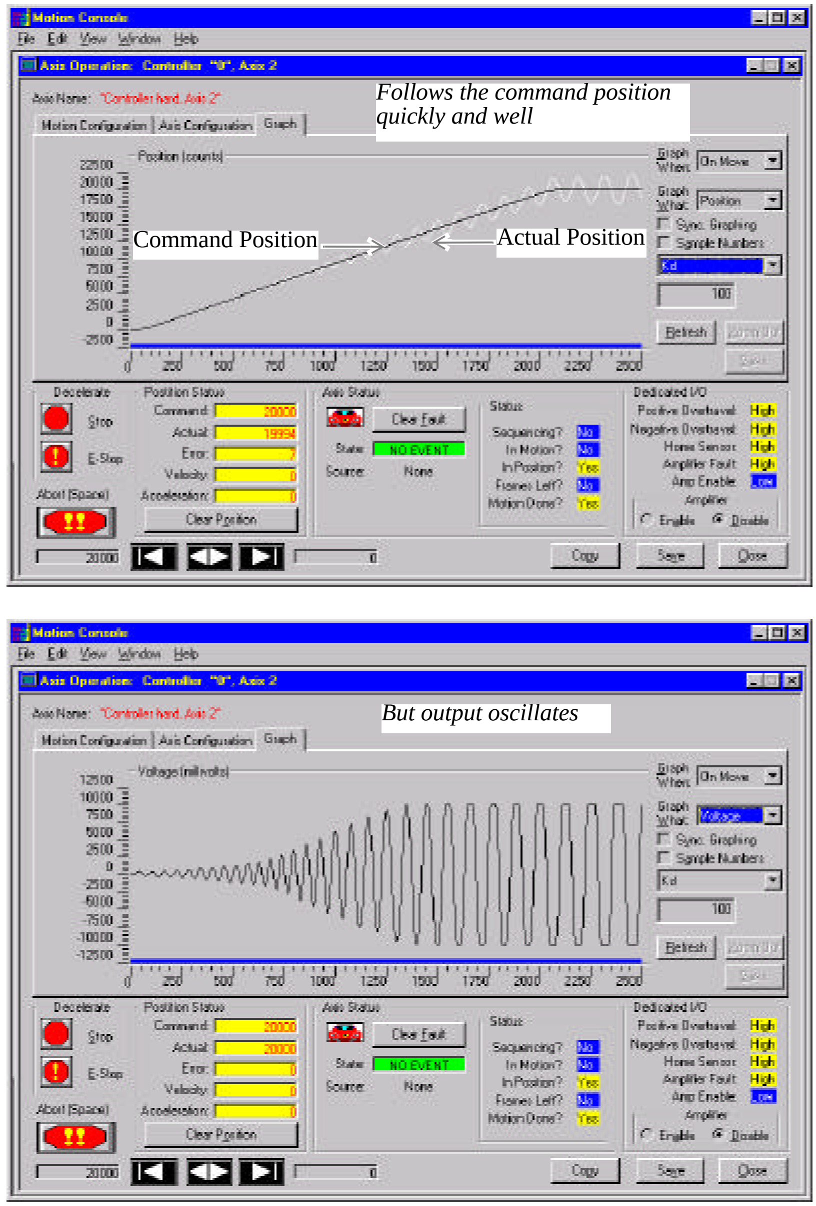
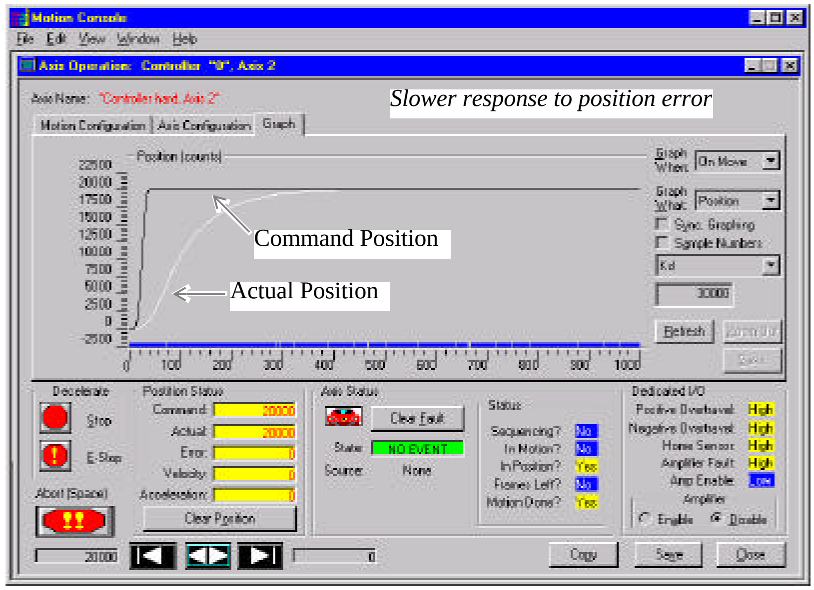
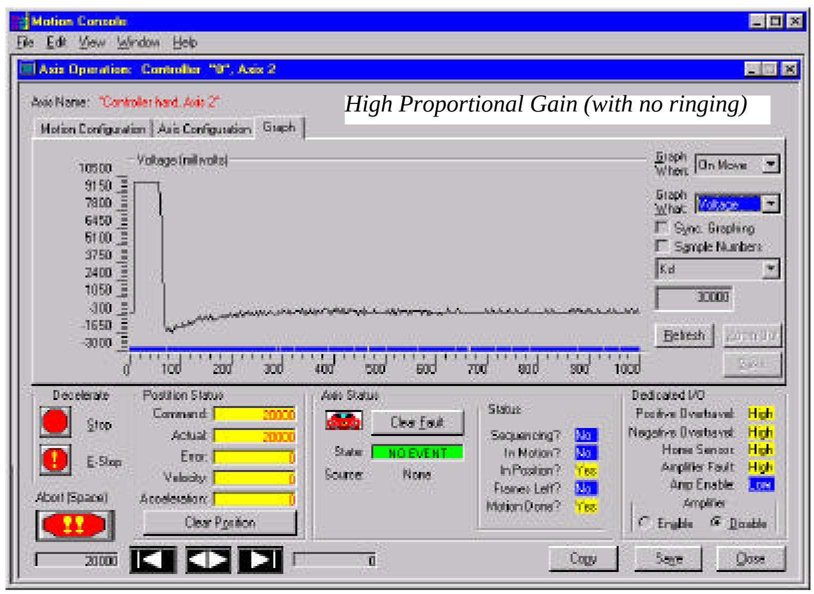


Figure D-6 Excessive Derivative Gain

HIGH DERIVATIVE GAIN





Integral Gain (Ki)

The Integral Gain increases/decreases the motor control output signal, based on the *summation of position error* as a function of time. Integral Gain helps the control system overcome static position errors caused by friction or loading.

Table D-7 Effects of Integral Gain

If Integral Gain is	*System Response*
Low or zero	has position errors at rest
Higher	has smaller position errors at rest but may “hunt” for the desired position

A low or zero value for Integral Gain may have *position errors* at rest, which depend on the static or frictional loads and the Proportional Gain. Increasing the Integral Gain can reduce these errors. If the Integral Gain is too large, the system may “hunt” (oscillate at low frequency) about the desired position.

Table D-8 Typical Integral Gain Values

	Typical Integral Gain Values are
	0-32, depending on the Integration Limit

Table D-9 Integral Mode Configurations

If Integral Mode is Configured for	*The Integration term is*
Only Standing	*only applied* when the command velocity is zero (recommended)
Always	*always* applied. (The summation of position error can be limited with the Integration Limit.)

Figure D-7 Insufficient Integral Gain (only when standing)

LOW INTEGRAL GAIN

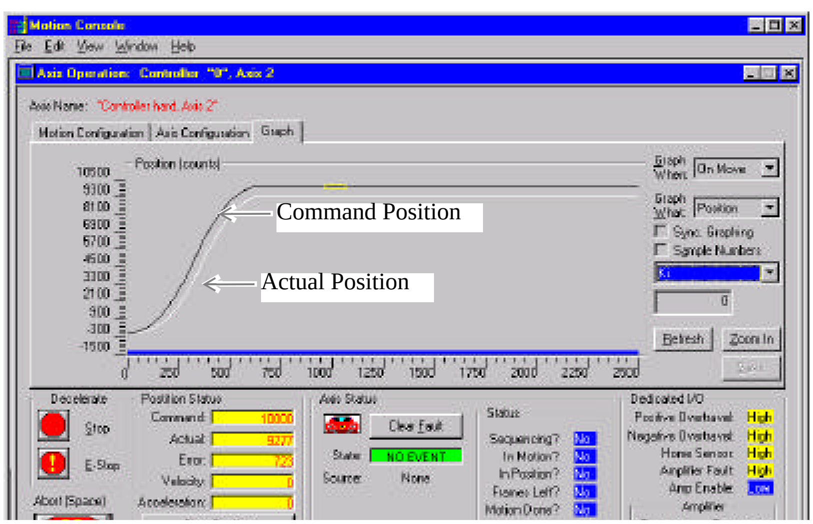
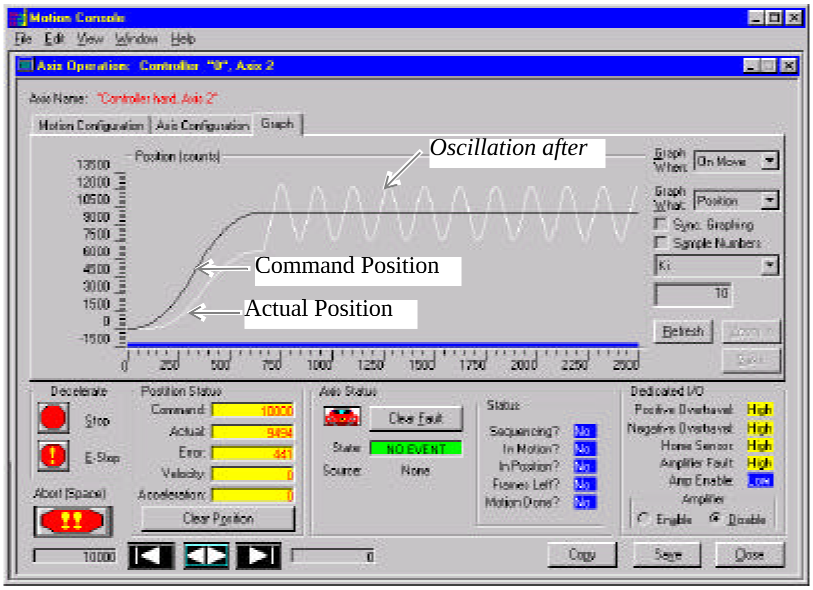


Figure D-8 High Integral Gain (only when standing)

HIGH INTEGRAL GAIN



Velocity Feed-Forward (Kv)

The Velocity Feed-Forward increases/decreases the motor control output signal, based on the *command velocity*. The Velocity Feed-Forward term is very important when used with **velocity-controlled servos** or **closed-loop step motors**.

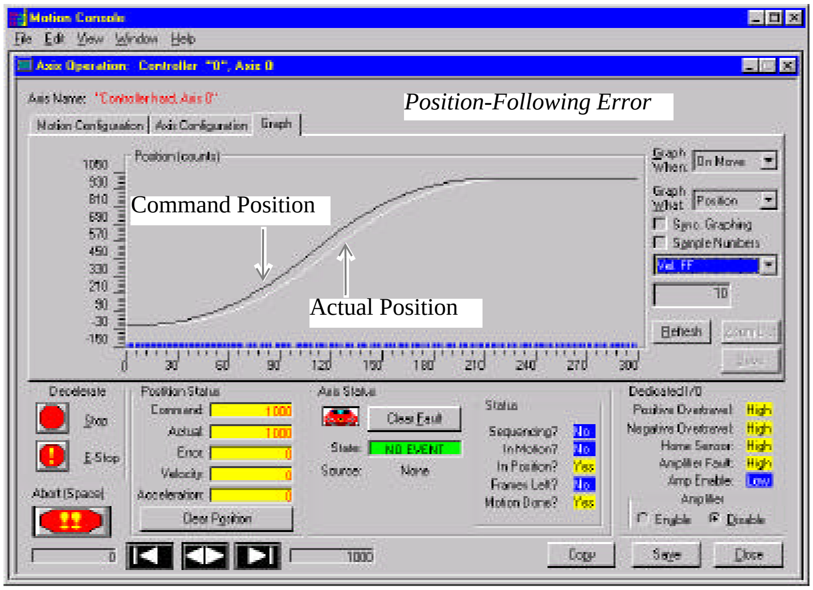
As the *command velocity* increases, the *position error* increases and a higher output voltage or pulse rate is needed to reduce the *following-position error*. The Velocity Feed-Forward term reduces the *following position error* by increasing the controller output voltage proportionally to the command velocity.

Table D-10 Effects of Velocity Feed-Forward

If Velocity Feed-Forward is	*Then*
too large	the motor will try to travel ahead of the command position
too small	the system will incur a position-following error

Figure D-9 Insufficient Velocity Feed-Forward

LOW VELOCITY FEED-FORWARD



Acceleration Feed-Forward (Ka)

The Acceleration Feed-Forward Gain (Ka) increases/decreases the motor control output signal, based on the *acceleration rate*. Acceleration Feed-Forward is used with torque-controlled servos (current). Systems with large inertial loads need more motor current to accelerate or decelerate than systems with light loads do. The Acceleration Feed-Forward Gain causes the controller to increase the motor control signals *during periods of* acceleration and deceleration.

Offset (Ko)

The Offset Gain term adds/subtracts a fixed value to or from the motor control output signal. You typically use the Offset Gain to compensate for small variations in controller DAC outputs and amplifier offsets, or to compensate for a fixed force (such as gravity) that is applied externally to a control system. Note that the internal offset in a DSP controller is **calibrated at the factory**, so that when the offset is zero, the analog or pulse output is also zero.

If necessary, use the CONFIG program to re-calibrate the analog and step pulse output.

Scale

You use the Scale parameter to adjust the resolution of the PID and feed-forward terms. The Scale parameter is used to calculate the overall scale factor K_R ($K_R = 2^{scale}$). The overall scale factor scales the *other* tuning parameters.

Decreasing the Scale by 1 will divide the equation for On by a factor of 2. In order to get the same voltage output from the PID, the gains and feed-forward terms must be doubled, i.e., a gain of 10 must be changed to a gain of 20. This means that a gain of 9.5 (that before could not have been entered) can now be entered as 19.

Friction Feed-Forward

The Friction Feed-Forward parameter adds a constant value to the DAC output when the command velocity is non-zero. The sign of the value applied to the DAC is equal to the sign of the command velocity multiplied by the friction feed-forward term. The Friction Feed-Forward term is 16-bits and has a range from -32,768 to 32,767. Generally, torque-controlled motion systems with constant friction benefit most from using a friction feed-forward term.

Integration Limit

The integrator sums the position error as a function of time. The integrator summation is limited using the Integration Limit. This prevents the integrator from building up a large position error summation and saturating the motor control output signal.

Use the Integration Limit with systems that have very high static friction.

Tuning Closed-Loop Servos

To quickly tune a stable system **with minimal position errors**:

Step 1: Set Proportional Gain
Step 2: Set Derivative Gain
Step 3: Iterate steps 1 and 2
Step 4: Set Integral Gain
Step 5: Set Velocity and Acceleration Feed-Forward

For new systems. **perform this sequence of steps twice**:
first *with no motor load* to provide a stable set of starting terms;
second *with the motor loaded* to fine-tune the initial parameters.

Step 1: Set Proportional Gain (Kp)

Start with all of the gains (except the offset Ko) set to 0 (Kp, Kd, Ki, Kv, and Ka). The motor should not turn and the shaft should be free (for *torque mode* servo drives). If the shaft turns, adjust the amplifier offset to reduce the motor speed to zero.

Set the Proportional Gain (Kp) to 1. Watch the position error on the *Motion Graph* window as the gain is changed. The *position error* should decrease as the Proportional Gain is increased.

If the motor runs away or the shaft still turns freely, verify the wiring.

Increase the gain by factors of two until the system begins to hum or oscillate. Reduce the Proportional Gain to half the value that first produces oscillation. The stability can be tested by physically "bumping" the motor shaft or mechanical system. An external impulse should not cause the motor to oscillate if the Proportional Gain (Kp) is set properly.

Step 2: Set the Derivative Gain

Start with a Derivative Gain (Kd) equal to the Proportional Gain (Kp). Increase the value of Kd by factors of two. Set the Derivative Gain (Kd) value to the smallest value which produces *no overshoot* during a two-point motion.

Step 3: Iterate Steps 1 and 2

With a Derivative Gain high enough to eliminate overshoot, increase the Proportional Gain until the system becomes unstable.

Now increase the Derivative Gain again and try to reduce overshoot and ringing. Eventually it will be impossible to eliminate the overshoot by raising Derivative Gain. At this point, the Proportional Gain should be reduced to provide the desired motion response. Remember that some overshoot is acceptable in systems which are being tuned for maximum speed.

Step 4: Set Integral Gain (Ki)

Observing the static error at the end of a move *as the Ki term is increased* is the best way to tune the Integral Gain. Using the two-point motion window, set the following motion parameters:

Table D-11 Integral Gain (Ki) Values for Tuning Closed-Loop Servos

Parameter	*Value*
Delay	2
Position 1	0
Position 2	20000
Velocity	10000
Acceleration	10000

Start the motion and observe the position error between moves. Gradually increase the Integral Gain (Ki) until the final position error is minimized. As you increase the Integral Gain above this level, watch for oscillation at the beginning or end of the motion. If oscillation at the beginning or end of the motion occurs, reduce the value of the Integral Gain.

Step 5: Set Velocity and Acceleration Feed-Forward

Using the fields in the *Movement* and *Motion Parameters* controls, specify a move that takes 5 to 10 seconds using the highest desired speed and acceleration.

Notice the position error during the *constant speed* portion of the motion. Increase the Velocity Feed-Forward (Kv) until the *constant velocity error* is minimized.

Use the same process to adjust the Acceleration Feed-Forward (Ka), watching the acceleration error during the acceleration and deceleration portions of the motion (look quickly if the acceleration time is short). Increase the Acceleration Feed-Forward until the *constant acceleration error* is minimized.

Tuning Closed-Loop Steppers

Warning!	For best performance, be sure the ratio between the encoder resolution (counts per rev) and the step resolution (steps or microsteps per rev) is 1:4. Lower ratios (1:1, 1:2) will be difficult to tune and will have poor static stability. Higher ratios (1:6, 1:8, etc.) will have poor constant velocity stability.

To tune a stable system with *minimal position errors*:

Step 1: Set Proportional Gain
Step 2: Set Velocity Feed-Forward & Acceleration Feed-Forward Gains
Step 3: Set Integral Gain

For new systems. **perform this sequence of steps twice**:
first *with no motor load* to provide a stable set of starting terms;
second *with the motor loaded* to fine-tune the initial parameters.

Step 1: Set Proportional Gain (Kp)

The Proportional Gain is dependent upon the ratio between the *number of encoder counts* and the *number of steps (or microsteps) per revolution* of the motor. The greater the number of steps per encoder count, the larger the Proportional Gain. Typically, the Proportional Gain will be between 20 and 400.

Start with the Proportional Gain at 20 and all other gains at 0 (Kd, Ki, Kv, and Ka).

Try some two-point motions and increase the Proportional Gain until the motor stalls. Then reduce the Proportional Gain to half the value (of Kp) that caused the motor to stall.
Be sure to write down this Kp value.

Now reduce the Proportional Gain to a very small value (about 1/10 of the current value).

Step 2: Set Velocity & Acceleration Feed-Forward Gains (Kv, Ka)

Using the fields in the *Movement* and *Motion Parameters* controls, specify a move with a typically desired speed and acceleration, and that also takes 5 to 10 seconds to complete.

Notice the position error during the *constant speed* portion of the motion. Increase the Velocity Feed-Forward Gain (Kv) until the *constant velocity error* is minimized. An optimum Kv gain is very important for closed-loop stepper systems.

Use the same method to adjust the Acceleration Feed-Forward Gain (Ka), watching the *acceleration error* during the acceleration and deceleration portions of the motion. Note that the Acceleration Feed-Forward won't be needed for most systems (but tune it anyway).

After the Velocity and Acceleration Feed-Forward Gains are set, increase the Proportional Gain back to the value you recorded during Step 1.

Step 3: Set the Integral Gain (Ki)

Observing the static error at the end of a move *as the Integral Gain (Ki) term is increased* is the best way to tune the Integral Gain. Using the two-point motion window, set the following motion parameters:

Table D-12 Integral Gain (Ki) Values for Tuning Closed-Loop Steppers

Parameter	***Value***
Delay	2
Position 1	0
Position 2	20000
Velocity	10000
Acceleration	10000

Start the motion and observe the *position error* between moves. Gradually increase the Integral Gain (Ki) until the *final position error* is minimized.

As you increase the Integral Gain (Ki) above this level, watch for oscillation *at the beginning or end of the motion*. If oscillation occurs at the beginning or end of the motion, reduce the value of the Integral Gain (Ki).

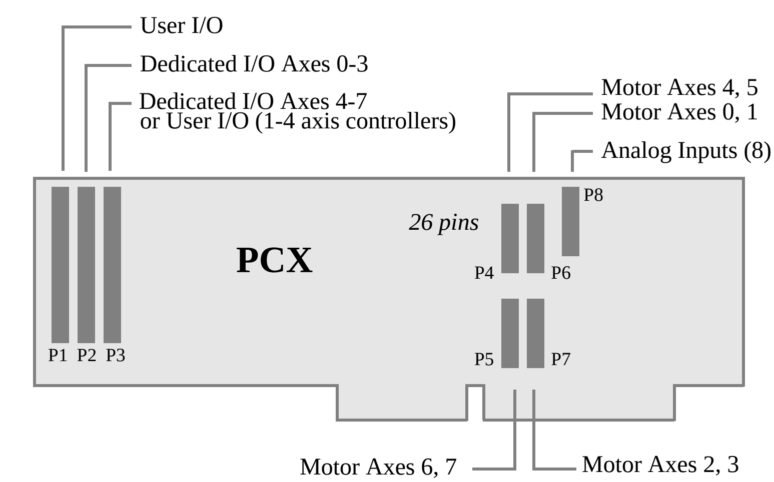
Step 3: Set the Integral Gain (Ki)

APPENDIX E — CONNECTIONS & SPECIFICATIONS

Motor Signal Header Locations

PCX

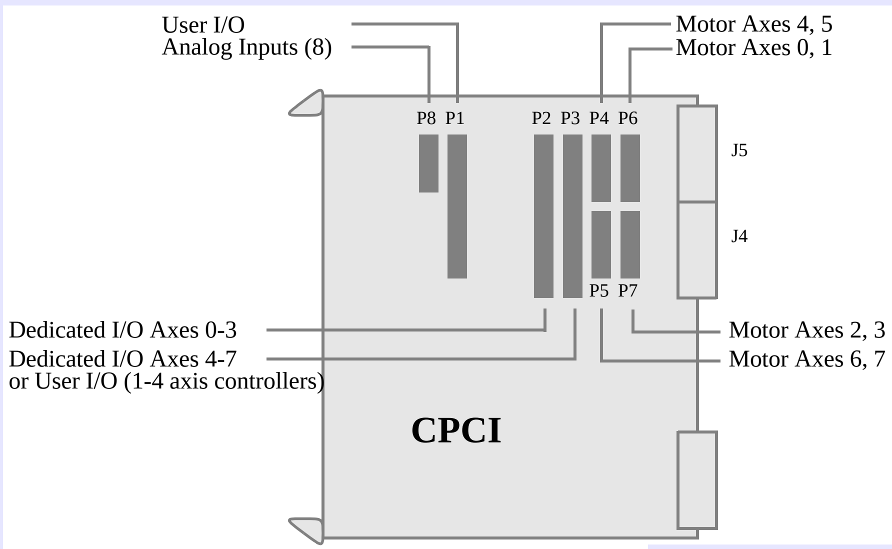
Figure E-1 Motor Signal Header Locations - PCX



PCX - Motor Signal Headers

CPCI

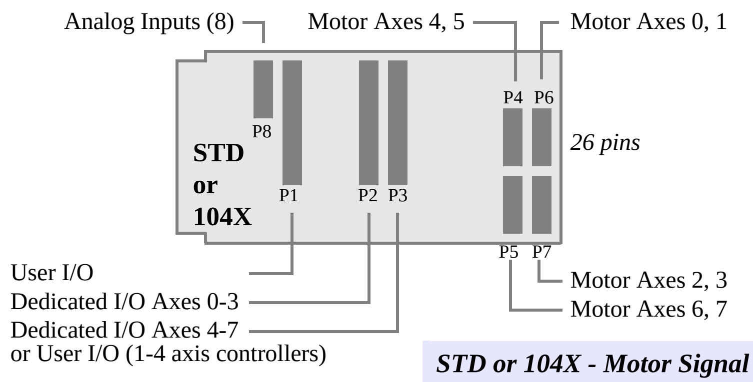
Figure E-2 Motor Signal Header Locations - CPCI



CPCI - Motor Signal Headers

STD, 104X

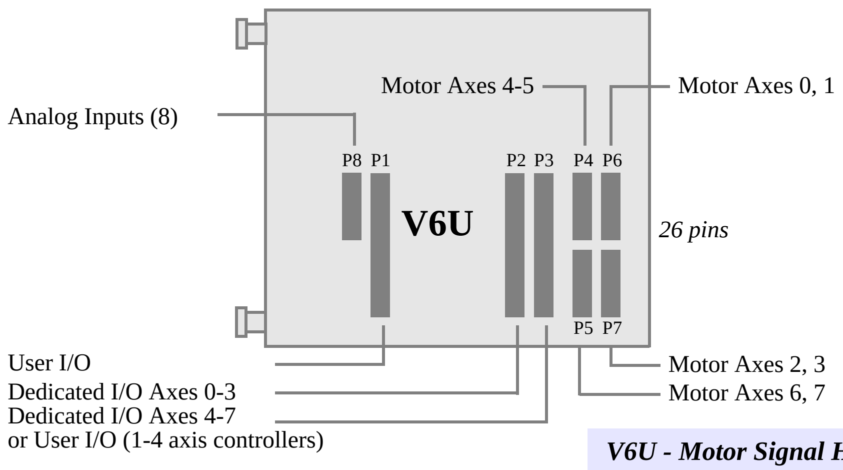
Figure E-3 Motor Signal Header Locations - STD or 104X



STD or 104X - Motor Signal Headers

V6U

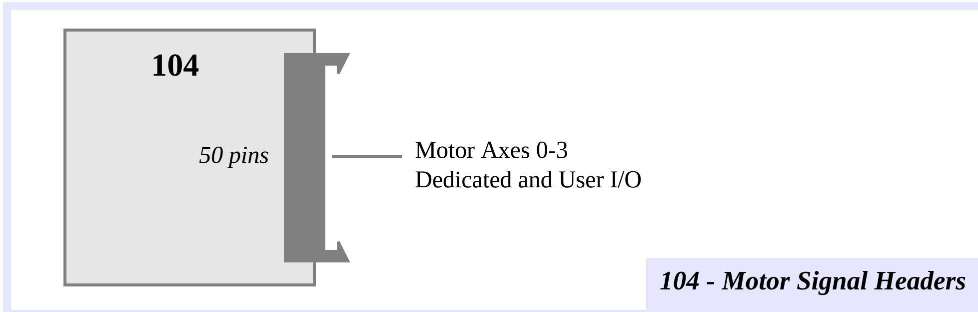
Figure E-4 Motor Signal Header Locations - V6U



V6U - Motor Signal Headers

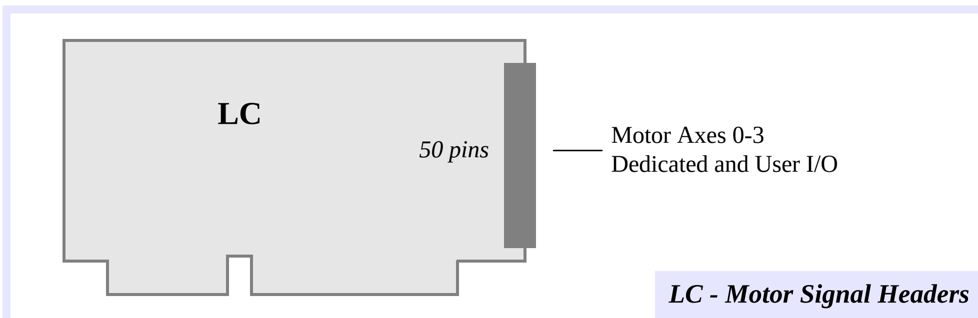
104

Figure E-5   Motor Signal Header Locations - 104



LC

Figure E-6   Motor Signal Header Locations - LC



Dedicated & User I/O

The DSP Series products have discrete digital I/O lines divided into 2 groups: **Dedicated** I/O and **User** I/O.

Dedicated I/O

There are 6 Dedicated I/O signals for each axis of the controller, 4 inputs and 2 outputs:

Inputs: Positive Limit
Negative Limit
Home
Amplifier Fault

Outputs: In Position
Amplifier Enable

User I/O

The PCX, CPCI, STD, and V6U support 24 or 44-bits of general purpose user I/O. The 104 and LC support 20-bits of general purpose user I/O. The PCI supports 24-bits of of general purpose User I/O. These signals can be configured as inputs or outputs in groups of 8.

Some restrictions apply. Dedicated I/O for axes 4-7 is available for User I/O on PCX, CPCI, STD, and V6U controllers with 4 or less axes. If an 8-bit port with Home sensor signals is configured as an output port, only 6 of the 8 signals can be used.

For the PCX, CPCI, STD, and V6U the Home Sensor inputs are not available for User I/O. The following diagram shows the configuration of the 72 I/O lines for the PCX, CPCI, STD, and V6U.

PCX, CPCI, STD & V6U

Figure E-7 User & Dedicated I/O Headers - PCX, CPCI, STD, and V6U

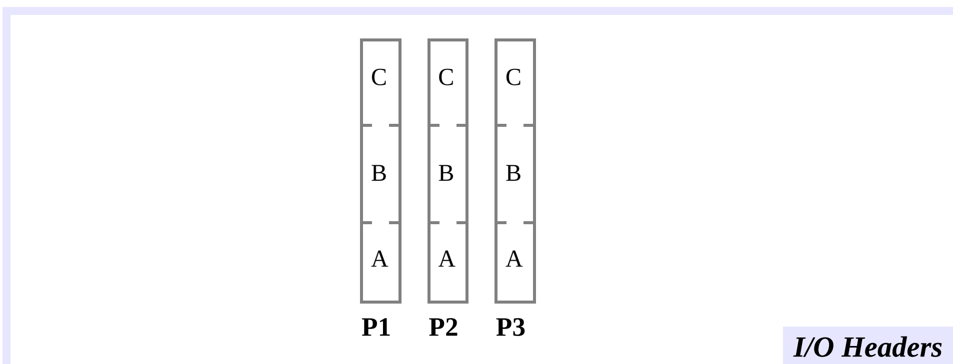


Table E-1 User I/O Headers

I/O	*Description*
P1A	User Port 0 (8-bits input or output)
P1B	User Port 1 (8-bits input or output)
P1C	User Port 2 (8-bits input or output)
P2A	Dedicated Inputs for Axes 0 and 1
P2B	Dedicated Inputs for Axes 2 and 3
P2C	Dedicated Outputs for Axes 0-3
P3A	Dedicated Inputs for Axes 4 and 5 or User Port 3 (6-bits in or 6-bits out)*
P3B	Dedicated Inputs for Axes 6 and 7 or User Port 4 (6-bits in or 6-bits out)*
P3C	Dedicated Outputs for Axes 4-7 or User Port 5 (8-bits in or 6-bits out)*

Table Notes	**The function of the signals on P3 depends on the number of axes:* • *Controllers with 5 or more axes use P3 for Dedicated I/O.* • *On controllers with 4 axes or less, P3 is available for User I/O.* *If P3 is used for User I/O and is a controller with 4 or less axes, then User Ports 3 and 4 can be configured for 6 inputs or 6 outputs.*

For PCX, STD,V6U, 104X, and CPCI DSP Series controllers: User I/O connections, 50 pin headers, Opto-22, Grayhill/Gordos-compatible, *even-numbered pins are grounds* and *pin 49 is +5V.*

Table E-2 User I/O Available on PCX/STD/V6U/104X/CPCI Controllers

Bit	Port	Header	Pin	Bit	Port	Header	Pin	Bit	Port	Header	Pin
0	0	P1	47	8	1	P1	31	16	2	P1	15
1	0	P1	45	9	1	P1	29	17	2	P1	13
2	0	P1	43	10	1	P1	27	18	2	P1	11
3	0	P1	41	11	1	P1	25	19	2	P1	9
4	0	P1	39	12	1	P1	23	20	2	P1	7
5	0	P1	37	13	1	P1	21	21	2	P1	5
6	0	P1	35	14	1	P1	19	22	2	P1	3**
7	0	P1	33	15	1	P1	17	23	2	P1	1***

Table E-3 User I/O Available on Controllers with 4 or Less Axes

Bit	Port	Header	Pin	Bit	Port	Header	Pin	Bit	Port	Header	Pin
24	3	P3	47	32	4	P3	31	40	5	P3	15
25	3	P3	45	33	4	P3	29	41	5	P3	13
26	3	P3	43	34	4	P3	27	42	5	P3	11
27	3	P3	41	35	4	P3	25	43	5	P3	9
28	3	P3	39	36	4	P3	23	44	5	P3	7
29	3	P3	37	37	4	P3	21	45	5	P3	5
30	3	P3	35	38	4	P3	19	46	5	P3	3
31	3	P3	33	39	4	P3	17	47	5	P3	1

Table Notes	** bits 26, 30, 34, and 38 can be configured as inputs on the PC/DSP, and are not available on the PCX, STD, V6U, 104X and CPCI cards with less than 5 axes.* ***bit 22 can also be used as "DSP Interrupt"* ****bit 23 can also be used as "PC Interrupt"*

PCI

The following table shows the configuration of the 24 User I/O lines for the PCI.

Figure E-8 User I/O Availabel on the PCI

Bit	*Description*	*Pin*
0	User Port A (8-bits input or output)	1
1		2
2		3
3		4
4		5
5		6
6		7
7		8

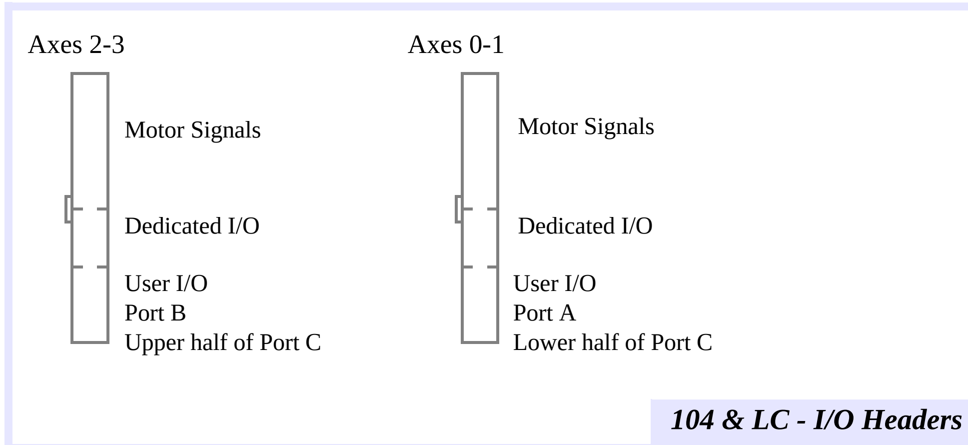
Bit	*Description*	*Pin*
8	User Port B (8-bits input or output)	9
9		10
10		11
11		12
12		13
13		14
14		15
15		16

Bit	*Description*	*Pin*
16	User Port C (8-bits input or output)	17
17		18
18		19
19		20
20		21
21		22
22		23
23		24

Both Dedicated and User I/O signals originate from 82C55 programmable I/O controllers. These signals can be programmed in groups of 8 as inputs or outputs.

104, LC

Figure E-9 User and Dedicated I/O Headers -104 & LC



The following table shows the configuration of the 20 User I/O lines for the LC and 104, as seen from the CBL-100 50-pin breakout cables:

Table E-4 104, LC - User I/O Connections, 100 pin connector

Bit	Port	Module	Pin
0	0	1	39
1	0	1	41
2	0	1	43
3	0	1	40
4	0	1	42
5	0	1	44
6*	0	-	-
7*	0	-	-
8	1	2	89
9	1	2	91
10	1	2	93
11	1	2	90
12	1	2	92
13	1	2	94
14*	-	-	-
15*	-	-	-
16	2	1	45
17	2	1	47
18	2	1	46
19	2	1	48
20	2	2	95
21	2	2	97
22	2	2	96**
23	2	2	98***

Table Notes	** bits 6, 7, 14 and 15 are not available on the 104 or LC.* *** bit 22 can also be used as "DSP Interrupt"* ****bit 23 can also be used as "PC Interrupt"*

Both Dedicated and User I/O signals come directly from 82C55 programmable I/O controllers. These signals can be programmed in groups of 8 as inputs or outputs. The input state is a high impedance TTL-level input. The output state has TTL-logic levels with +/-2.5 mA drive current (4.0 mA max). The power-up state of all User I/O is high impedance (input state).

Pinouts

PCX, CPCI, STD, 104X, V6U

Table E-5 Pinouts

Motor Axes Connections 26-pin box header		
Pin	***Signal***	***Axis***
1	GND	1st
2	5V	1st
3	Encoder A +	1st
4	Encoder A -	1st
5	Encoder B +	1st
6	Encoder B -	1st
7	Encoder Index +	1st
8	Encoder Index -	1st
9	+/- 10V Analog Out	1st
10	Step Pulse + *	1st
11	Step Pulse - *	1st
12	Step Direction + *	1st
13	Step Direction - *	1st
14	GND	2nd
15	5V	2nd
16	Encoder A +	2nd
17	Encoder A -	2nd
18	Encoder B +	2nd
19	Encoder B -	2nd
20	Encoder Index +	2nd
21	Encoder Index -	2nd
22	+/- 10V Analog Out	2nd
23	Step Pulse + *	2nd
24	Step Pulse - *	2nd
25	Step Direction + *	2nd
26	Step Direction - *	2nd

Analog Input Connections 20-pin box header (P8)	
Pin	***Signal***
1	GND
2	Analog GND
3	Clock 0
4	Analog in 0
5	-12V
6	Analog in 1
7	+12V
8	Analog in 2
9	+5V
10	Analog in 3
11	Gate 0
12	Analog in 4
13	Out 0
14	Analog in 5
15	Out 1
16	Analog in 6
17	Out 2
18	Analog in 7
19	GND
20	Analog GND

Note: Two motors of the same type can be connected to each motor header.

Table E-6 Dedicated and User I/O Connections

Dedicated I/O Connections 50-pin box headers			
Pin	**Signal**	**P2 Axis**	**P3 Axis**
1	In-Position Out	3	7
3	Amp Enable Out	3	7
5	In-Position Out	2	6
7	Amp Enable Out	2	6
9	In-Position Out	1	5
11	Amp Enable Out	1	5
13	In-Position Out	0	4
15	Amp Enable Out	0	4
17	Amp Fault Input	3	7
19	Home Input	3	7
21	NEG Limit Input	3	7
23	POS Limit Input	3	7
25	Amp Fault Input	2	6
27	Home Input	2	6
29	NEG Limit Input	2	6
31	POS Limit Input	2	6
33	Amp Fault Input	1	5
35	Home Input	1	5
37	NEG Limit Input	1	5
39	POS Limit Input	1	5
41	Amp Fault Input	0	4
43	Home Input	0	4
45	NEG Limit Input	0	4
47	POS Limit Input	0	4
49	5V		

Note: Even numbered pins are grounds and pin 49 is 5V

User I/O Connections 50-pin Opto-22 compatible header (P1)	
Pin	**Signal**
1	I/O Line C-7 or PC Interrupt
3	I/O Line C-6 or DSP Interrupt
5	I/O Line C-5
7	I/O Line C-4
9	I/O Line C-3
11	I/O Line C-2
13	I/O Line C-1
15	I/O Line C-0
17	I/O Line B-7
19	I/O Line B-6
21	I/O Line B-5
23	I/O Line B-4
25	I/O Line B-3
27	I/O Line B-2
29	I/O Line B-1
31	I/O Line B-0
33	I/O Line A-7
35	I/O Line A-6
37	I/O Line A-5
39	I/O Line A-4
41	I/O Line A-3
43	I/O Line A-2
45	I/O Line A-1
47	I/O Line A-0
49	5V

Note: Each I/O Port (A,B,C) can be defined as inputs or outputs

CPCI/DSP Rear I/O

Table E-7 J4 Rear I/O Connections

Pin	Signal	Pin	Signal
A1	User I/O PB4	B1	User I/O PB5
A2	User I/O PB0	B2	User I/O PB1
A3	User I/O PA4	B3	User I/O PA5
A4	User I/O PA0	B4	User I/O PA1
A5	Amp Enable(7)	B5	Amp Fault(7)
A6	In Position(6)	B6	*Reserved*
A7	Amp Enable(6)	B7	Amp Fault(6)
A8	Amp Enable(5)	B8	Amp Fault(5)
A9	In Position(4)	B9	*Reserved*
A10	Amp Enable(4)	B10	Amp Fault(4)
A11	Amp Enable(3)	B11	Amp Fault(3)
A12 to A14	Key location, no pins	B12 to B14	Key location, no pins
A15	In Position(2)	B15	*Reserved*
A16	Amp Enable(2)	B16	Amp Fault(2)
A17	Amp Enable(1)	B17	Amp Fault(1)
A18	In Position(0)	B18	*Reserved*
A19	Amp Enable(0)	B19	Amp Fault(0)
A20	Encoder Index(6) +	B20	Encoder Index(6) -
A21	Encoder A(7) +	B21	Encoder A(7) -
A22	Encoder A(6) +	B22	Encoder A(6) -
A23	Encoder A(5) +	B23	Encoder A(5) -
A24	Encoder A(4) +	B24	Encoder A(4) -
A25	Encoder Index(3) +	B25	Encoder Index(3) -

Table E-8 J4 Rear I/O Connections (Continued)

Pin	Signal	Pin	Signal
C1	User I/O PB6	D1	User I/O PB7
C2	User I/O PB2	D2	User I/O PB3
C3	User I/O PA6	D3	User I/O PA7
C4	User I/O PA2	D4	User I/O PA3
C5	Home Input(7)	D5	Positive Limit(7)
C6	User I/O PC3	D6	In Position(7)
C7	Home Input(6)	D7	Positive Limit(6)
C8	Home Input(5)	D8	Positive Limit(5)
C9	User I/O PC2	D9	In Position(5)
C10	Home Input(5)	D10	Positive Limit(4)
C11	Home Input(3)	D11	Positive Limit(3)
C12 to C14	Key location, no pins	D12 to D14	Key location, no pins
C15	User I/O PC1	D15	In Position(3)
C16	Home Input(2)	D16	Positive Limit(2)
C17	Home Input(1)	D17	Positive Limit(1)
C18	User I/O PC0	D18	In Position(1)
C19	Home Input(0)	D19	Positive Limit(0)
C20	Encoder Index(7) +	D20	Encoder Index(7) -
C21	Encoder B(7) +	D21	Encoder B(7) -
C22	Encoder B(6) +	D22	Encoder B(6) -
C23	Encoder B(5) +	D23	Encoder B(5) -
C24	Encoder B(4) +	D24	Encoder B(4) -
C25	Encoder Index(4) +	D25	Encoder Index(4) -

Table E-9 J4 Rear I/O Connections (Continued)

Pin	***Signal***	***Pin***	***Signal***
E1	User I/O PC7 or PC Interrupt	F1	GND
E2	User I/O PC6 or DSP Interrupt	F2	GND
E3	User I/O PC5	F3	GND
E4	User I/O PC4	F4	GND
E5	Negative Limit(7)	F5	GND
E6	*Reserved*	F6	GND
E7	Negative Limit(6)	F7	GND
E8	Negative Limit(5)	F8	GND
E9	*Reserved*	F9	GND
E10	Negative Limit(4)	F10	GND
E11	Negative Limit(3)	F11	GND
E12	Key location, no pins	F12	GND
to E14		F13	GND
E15	*Reserved*	F14	GND
E16	Negative Limit(2)	F15	GND
E17	Negative Limit(1)	F16	GND
E18	*Reserved*	F17	GND
E19	Negative Limit(0)	F18	GND
E20	*Reserved*	F19	GND
E21	*Reserved*	F20	GND
E22	Encoder Index(5) -	F21	GND
E23	Encoder Index(5) +	F22	GND
E24	GND	F23	GND
E25	+5V	F24	GND
		F25	GND

Table E-10 J5 Rear I/O Connections

Pin	***Signal***	***Pin***	***Signal***
A1	Encoder A(3) +	B1	Encoder A(3) -
A2	Encoder A(2) +	B2	Encoder A(2) -
A3	Encoder A(1) +	B3	Encoder A(1) -
A4	Encoder A(0) +	B4	Encoder A(0) -
A5	Direction(6) +	B5	Direction(6) -
A6	Direction(5) +	B6	Direction(5) -
A7	Direction(4) +	B7	Direction(4) -
A8	Direction(3) +	B8	Direction(3) -
A9	Direction(2) +	B9	Direction(2) -
A10	Direction(1) +	B10	Direction(1) -
A11	Direction(0) +	B11	Direction(0) -
A12	Clock 0	B12	Gate 0
A13	+/- 10V Analog Out(7)	B13	Analog Out Ref(7)
A14	+/- 10V Analog Out(6)	B14	Analog Out Ref(6)
A15	+/- 10V Analog Out(5)	B15	Analog Out Ref(5)
A16	+/- 10V Analog Out(4)	B16	Analog Out Ref(4)
A17	+/- 10V Analog Out(3)	B17	Analog Out Ref(3)
A18	+/- 10V Analog Out(2)	B18	Analog Out Ref(2)
A19	+/- 10V Analog Out(1)	B19	Analog Out Ref(1)
A20	+/- 10V Analog Out(0)	B20	Analog Out Ref(0)
A21	Analog in 4	B21	Analog in 5
A22	Analog in 0	B22	Analog in 1

Table E-11 J5 Rear I/O Connections (Continued)

Pin	*Signal*	*Pin*	*Signal*
C1	Encoder B(3) +	D1	Encoder B(3) -
C2	Encoder B(2) +	D2	Encoder B(2) -
C3	Encoder B(1) +	D3	Encoder B(1) -
C4	Encoder B(0) +	D4	Encoder B(0) -
C5	Step Pulse(6) +	D5	Step Pulse(6) -
C6	Step Pulse(5) +	D6	Step Pulse(5) -
C7	Step Pulse(4) +	D7	Step Pulse(4) -
C8	Step Pulse(3) +	D8	Step Pulse(3) -
C9	Step Pulse(2) +	D9	Step Pulse(2) -
C10	Step Pulse(1) +	D10	Step Pulse(1) -
C11	Step Pulse(0) +	D11	Step Pulse(0) -
C12	Out 0	D12	Out 1
C13	GND	D13	*Reserved*
C14	+5V	D14	*Reserved*
C15	*Reserved*	D15	*Reserved*
C16	*Reserved*	D16	*Reserved*
C17	Analog GND	D17	*Reserved*
C18	+12V	D18	*Reserved*
C19	Analog GND	D19	*Reserved*
C20	-12V	D20	*Reserved*
C21	Analog GND	D21	Analog in 6
C22	Analog GND	D22	Analog in 2

Table E-12 J5 Rear I/O Connections (Continued)

Pin	*Signal*	*Pin*	*Signal*
E1	Encoder Index(2) -	F1	GND
E2	Encoder Index(2) +	F2	GND
E3	Encoder Index(1) -	F3	GND
E4	Encoder Index(1) +	F4	GND
E5	Encoder Index(0) -	F5	GND
E6	Encoder Index(0) +	F6	GND
E7	*Reserved*	F7	GND
E8	Step Pulse(7) -	F8	GND
E9	Step Pulse(7) +	F9	GND
E10	Direction(7) -	F10	GND
E11	Direction(7) +	F11	GND
E12	Out 2	F12	GND
E13	*Reserved*	F13	GND
E14	*Reserved*	F14	GND
E15	*Reserved*	F15	GND
E16	*Reserved*	F16	GND
E17	*Reserved*	F17	GND
E18	*Reserved*	F18	GND
E19	*Reserved*	F19	GND
E20	*Reserved*	F20	GND
E21	Analog in 7	F21	GND
E22	Analog in 3	F22	GND

Notes for CPCI Rear I/O Pinouts

Warning! The CPCI/DSP pin nomenclature follows the Compact PCI specification. Your connector manufacturer's documentation may use *different* nomenclature (typically rows 1-25 are reversed to be 25-1).

Analog Out Ref (0:7) are recommended as the reference signals for +/-10V *Analog Out (0:7)*. You may instead reference *GND*, as for previous MEI products.

Analog *GND* is recommended as the signal ground for *Analog In 7:0*.

Pins marked “Reserved” above should not be used, because these pins are reserved for future MEI functions.

Note that connector column Z is not shown (it does not connect to the CPCI, which does not have a bottom shield). Your backplane *may or may not* connect these to digital GND. It may be best to avoid connections to these pins.

PCI

Table E-13 **STC-136 Connection Module**

Axes (0,1)				*Axes (2,3)*			
Pin	***Signal***	***Pin***	***Signal***	***Pin***	***Signal***	***Pin***	***Signal***
1	Analog_0+	35	AGnd	1	Analog_6+	35	AGnd
2	Analog_1+	36	AGnd	2	Analog_7+	36	AGnd
3	Gnd	37	Gnd	3	Gnd	37	Gnd
4	Enc0_A+	38	Enc0_A-	4	Enc2_A+	38	Enc2_A-
5	Enc0_B+	39	Enc0_B-	5	Enc2_B+	39	Enc2_B-
6	Enc0_I+	40	Enc0_I-	6	Enc2_I+	40	Enc2_I-
7	Home0_IN	41	5V_OUT	7	Home2_IN	41	5V_OUT
8	Pos_Lim0_IN	42	Gnd_OUT	8	Pos_Lim2_IN	42	Gnd_OUT
9	Neg_Lim0_IN	43	Mech0_Rtn	9	Neg_Lim2_IN	43	Mech2_Rtn
10	Command_0+	44	Command_0-	10	Command_2+	44	Command_2-
11	*Reserved*	45	*Reserved*	11	*Reserved*	45	*Reserved*
12	Amp_Flt0_IN	46	Amp_Flt0_Rtn	12	Amp_Flt2_IN	46	Amp_Flt2_Rtn
13	Amp_En0_C	47	Amp_En0_E	13	Amp_En2_C	47	Amp_En2_E
14	*Reserved*	48	*Reserved*	14	*Reserved*	48	*Reserved*
15	Step0+	49	Step0-	15	Step2+	49	Step2-
16	Dir0+	50	Dir0-	16	Dir2+	50	Dir2-
17	In_Pos0+	51	In_Pos0-	17	In_Pos2+	51	In_Pos2-
18	Enc1_A+	52	Enc1_A-	18	Enc3_A+	52	Enc3_A-
19	Enc1_B+	53	Enc1_B-	19	Enc3_B+	53	Enc3_B-
20	Enc1_I+	54	Enc1_I-	20	Enc3_I+	54	Enc3_I-
21	Home1_IN	55	5V_OUT	21	Home3_IN	55	5V_OUT
22	Pos_Lim1_IN	56	Gnd_OUT	22	Pos_Lim3_IN	56	Gnd_OUT
23	Neg_Lim1_IN	57	Mech1_Rtn	23	Neg_Lim3_IN	57	Mech3_Rtn
24	Command_1+	58	Command_1-	24	Command_3+	58	Command_3-
25	*Reserved*	59	*Reserved*	25	*Reserved*	59	*Reserved*
26	Amp_Flt1_IN	60	Amp_Flt1_Rtn	26	Amp_Flt3_IN	60	Amp_Flt3_Rtn
27	Amp_En1_C	61	Amp_En1_E	27	Amp_En3_C	61	Amp_En3_E
28	Gnd	62	Gnd	28	Gnd	62	Gnd
29	Step1+	63	Step1-	29	Step3+	63	Step3-
30	Dir1+	64	Dir1-	30	Dir3+	64	Dir3-
31	In_Pos1+	65	In_Pos1-	31	In_Pos3+	65	In_Pos3-
32	Gnd	66	Gnd	32	Gnd	66	Gnd
33	Analog_4+	67	AGnd	33	Analog_2+	67	AGnd
34	Analog_5+	68	AGnd	34	Analog_3+	68	AGnd

Table E-14 **STC-D50 (User I/O Connector)**

Pin	*Signal*	*Pin*	*Signal*
1	UserIO_A0	26	UserIO_A0_Rtn
2	UserIO_A1	27	UserIO_A1_Rtn
3	UserIO_A2	28	UserIO_A2_Rtn
4	UserIO_A3	29	UserIO_A3_Rtn
5	UserIO_A4	30	UserIO_A4_Rtn
6	UserIO_A5	31	UserIO_A5_Rtn
7	UserIO_A6	32	UserIO_A6_Rtn
8	UserIO_A7	33	UserIO_A7_Rtn
9	UserIO_B0	34	UserIO_B0_Rtn
10	UserIO_B1	35	UserIO_B1_Rtn
11	UserIO_B2	36	UserIO_B2_Rtn
12	UserIO_B3	37	UserIO_B3_Rtn
13	UserIO_B4	38	UserIO_B4_Rtn
14	UserIO_B5	39	UserIO_B5_Rtn
15	UserIO_B6	40	UserIO_B6_Rtn
16	UserIO_B7	41	UserIO_B7_Rtn
17	UserIO_C0	42	UserIO_C0_Rtn
18	UserIO_C1	43	UserIO_C1_Rtn
19	UserIO_C2	44	UserIO_C2_Rtn
20	UserIO_C3	45	UserIO_C3_Rtn
21	UserIO_C4	46	UserIO_C4_Rtn
22	UserIO_C5	47	UserIO_C5_Rtn
23	UserIO_C6	48	UserIO_C6_Rtn
24	UserIO_C7	49	UserIO_C7_Rtn
25	5V	50	Gnd

104, LC

Table E-15 Connector Module 1 (Axes 0, 1) (Upper Cable)

Pin	Signal	Pin	Signal
1	+5V	2	+5V
3	Encoder A(0) +	4	Encoder A(1) +
5	Encoder A(0) -	6	Encoder A(1) -
7	Encoder B(0) +	8	Encoder B(1) +
9	Encoder B(0) -	10	Encoder B(1) -
11	Encoder Index(0) +	12	Encoder Index(1) +
13	Encoder Index(0) -	14	Encoder Index(1) -
15	+/- 10V Analog Out(0)	16	+/- 10V Analog Out(1)
17	GND	18	GND
19	Step Pulse(0) +	20	Step Pulse(1) +
21	Step Pulse(0) -	22	Step Pulse(1) -
23	Direction(0) +	24	Direction(1) +
25	Direction(0) -	26	Direction(1) -
27	Positive Limit(0)	28	Positive Limit(1)
29	Negative Limit(0)	30	Negative Limit(1)
31	Home Input(0)	32	Home Input(1)
33	Amp Fault(0)	34	Amp Fault(1)
35	Amp Enable(0)	36	Amp Enable(1)
37	In Position(0)	38	In Position(1)
39	User I/O PA0	40	User I/O PA3
41	User I/O PA1	42	User I/O PA4
43	User I/O PA2	44	User I/O PA5
45	User I/O PC0	46	User I/O PC2
47	User I/O PC1	48	User I/O PC3
49	GND	50	GND

Table E-16 Connector Module 2 (Axes 2, 3) (Lower Cable)

Pin	*Signal*	*Pin*	*Signal*
1	+5V	2	+5V
3	Encoder A(2) +	4	Encoder A(3) +
5	Encoder A(2) -	6	Encoder A(3) -
7	Encoder B(2) +	8	Encoder B(3) +
9	Encoder B(2) -	10	Encoder B(3) -
11	Encoder Index(2) +	12	Encoder Index(3) +
13	Encoder Index(2) -	14	Encoder Index(3) -
15	+/- 10V Analog Out(2)	16	+/- 10V Analog Out(3)
17	GND	18	GND
19	Step Pulse(2) +	20	Step Pulse(3) +
21	Step Pulse(2) -	22	Step Pulse(3) -
23	Direction(2) +	24	Direction(3) +
25	Direction(2) -	26	DIrection(3) -
27	Positive Limit(2)	28	Positive Limit(3)
29	Negative Limit(2)	30	Negative Limit
31	Home Input(2)	32	Home Input(3)
33	Amp Fault(2)	34	Amp Fault(3)
35	Amp Enable(2)	36	Amp Enable(3)
37	In Position(2)	38	In Position(3)
39	User I/O PB0	40	User I/O PB3
41	User I/O PB1	42	User I/O PB4
43	User I/O PB2	44	User I/O PB5
45	User I/O PC4	46	User I/O PC6 or DSP Interrupt
47	User I/O PC5	48	User I/O PC7 or PC Interrupt
49	GND	50	GND

Specifications

Power Consumption Notes

For power consumption specifications of a specific DSP Series model, refer to the tables on subsequent pages.

Maximum current requirements (IEEE P996 spec.) for 8-bit PC add-on cards are:

+5V..........3.0 amp
+12V........1.5 amp
-12V..........0.3 amp

The current dissipation for all DSP Series controllers follow:

Table E-17 Current dissipation for all DSP Series boards

	+5V (typical)	*+5V (max)*	*+12V (max)*	*-12V (max)*
4-axis board	0.513 amp	0.539 amp	0.004 amp	0.014 amp
8-axis board	0.609 amp	0.659 amp	0.008 amp	0.018 amp

The +5V, +12V, and -12V supply pins are brought out directly from the bus, which is connected to the backplane power supply. On each board, copper planes are used for 5V and GND. 12V power is through 25mil traces of 1oz. copper. (Allows 1 amp to flow with 10 degrees C temp rise.)

A *conservative estimate* of current that can be drawn off the supply pins of the on-board headers:

	4-axis controller	*8-axis controller*
+5V	600mA	200mA
+12V	500mA	500mA
-12V	200mA	150mA

A *typical estimate* of current that can be drawn off the supply pins of the on-board headers:

	4-axis controller	*8-axis controller*
+5V	1.3 amp	1.1 amp
+12V	500mA	500mA
-12V	200mA	150mA

PCX

Interface	PC/XT/AT-compatible Switch-selectable base address, I/O mapped Switch-selectable interrupts
Servo Loop Update Rate	10.0 kHz (1 axis) 3.0 kHz (4-axes simultaneously, maximum) 1.6 kHz (8-axes simultaneously, maximum) 1.25 kHz (default) User-programmable
Servo Output	± 10V DC @ 16-bit resolution (from 18-bit conversion) ± 18 mA current 100 ppm long-term velocity accuracy
Step Output	Maximum Step Frequency: 325 kHz RS-422 line driver outputs, ± 20 mA current 50% Duty Cycle Non-linearity < 1% at Full Scale
Ranges	Position: 32-bit, ±2.15 billion counts (steps) Velocity: 48-bit (±65 million counts/sec and 2 kHz sampling) Acceleration: 48-bit (±131 billion counts/sec^2 at 2 kHz sampling) Jerk: 48-bit (262 trillion counts/sec^3 at 2 kHz sampling)
Position Feedback	Input Frequency: 5 MHz (max) including Quadrature Quadrature, single-ended or differential (A,B,I) Digital Noise Filtering RS-422 Line receiver inuts 4.0 mA max current output
Motion Profiles	Trapezoidal, Parabolic, S-Curve acceleration & deceleration
Dedicated I/O	TTL-compatible, 4.0 mA drive on outputs No pull-up resistors are included
Dedicated Inputs (per axis)	Forward Limit (POS), Reverse Limit (NEG), Home, Amp-Fault
Dedicated Outputs (per axis)	In-Position Amp-Enable
User I/O	2/4 axis models = 44 lines, 6/8 axis models = 24 lines TTL compatible, 4.0 mA drive on outputs Direct access from Host PC
Analog Inputs	8 Channels @ 12-bit resolution Configurable for 4-channel differential mode 75 kHz sampling rate 5V Unipolar input, ±2.5V Bipolar input Direct access from Host processor
Power Requirements	*8 axis* / *4 axis* +5V Icc = .7 A max .6A max +12V Icc = 8mA max 4mA max -12V Icc = 18mA max 14mA max
Environmental Conditions	0 - 60 degrees C 32 - 140 degrees F 20 - 95% RH, non-condensing
Construction	Full SMT; 4-layer PCB 100% bed of nails and fully functionally tested with 24-hour burn-in

CPCI

Interface	CompactPCI 1.0-compatible Single 6U CompactPCI slot
Servo Loop Update Rate	10.0 kHz (1 axis) 3.0 kHz (4-axes simultaneously, maximum) 1.6 kHz (8-axees simultaneously, maximum) 1.25 kHz (default) User-programmable
Servo Output	± 10V DC @ 16-bit resolution (from 18-bit conversion) ± 18 mA current 100 ppm long-term velocity accuracy
Step Output	Maximum Step Frequency: 325 kHz RS-422 line driver outputs, ± 20 mA current 50% Duty Cycle Non-linearity < 1% at Full Scale
Ranges	Position: 32-bit, ±2.15 billion counts (steps) Velocity: 48-bit (±65 million counts/sec and 2 kHz sampling) Acceleration: 48-bit (±131 billion counts/sec^2 at 2 kHz sampling) Jerk: 48-bit (262 trillion counts/sec^3 at 2 kHz sampling)
Position Feedback	Input Frequency: 5MHz (max) including Quadrature Quadrature, single-ended or differential (A,B,I) Digital Noise Filtering RS-422 Line receiver inputs 4.0 mA max current input
Motion Profiles	Trapezoidal, Parabolic, S-Curve acceleration & deceleration
Dedicated I/O	TTL-compatible, 4.0 mA drive on outputs No pull-up resistors are included
Dedicated Inputs (per axis)	Forward Limit (POS), Reverse Limit (NEG), Home, Amp-Fault
Dedicated Outputs (per axis)	In-Position Amp-Enable
User I/O	2/4 axis models = 44 lines, 6/8 axis models = 24 lines TTL compatible, 4.0 mA drive on outputs Direct access from Host PC
Analog Inputs	8 Channels @ 12-bit resolution Configurable for 4-channel differential mode 75 kHz sampling rate 5V Uipolar input, ±2.5 Bipolar input Direct access from Host processor
Power Requirements	<u>8 axis</u> <u>4 axis</u> +5V Icc = .7 A max .6A max +12V Icc = 8mA max 4mA max -12V Icc = 18mA max 14mA max
Environmental Conditions	0 - 60 degrees C 32 - 140 degrees F 20 - 95% RH, non-condensing
Construction	Full SMT; 4-layer PCB 100% bed of nails and fully functionally tested with 24-hour burn-in

PCI

Interface	PCI-compatible Plug and Play addressing and IRQ selection
Servo Loop Update Rate	1.25 kHz (default) 2.7 kHz (4-axes simultaneously, maximum) 7.1 kHz (1 axis) User-programmable
Servo Output	±10V DC @ 16-bit resolution (from 18-bit conversion) ±18 mA current 100 ppm long-term velocity accuracy
Step Output	Maximum Step Frequency: 550 kHz RS-422 line driver outputs, ±20 mA current 50% Data Cycle Non-linearity < 1% at Full Scale
Ranges	Position: 32-bit, ±2.15 billion counts (steps) Velocity: 48-bit (±65 million counts/sec and 2 kHz sampling) Acceleration: 48-bit (±131 billion counts/sec^2 at 2 kHz sampling) Jerk: 48-bit (262 trillion counts/sec^3 at 2 kHz sampling)
Position Feedback	Input Frequency: 5 MHz (max) including Quadrature Quadrature, single-ended or differential (A,B,I) Digital Noise Filtering RS-422 line receiver inputs 4.0 mA max current input
Motion Profiles	Trapezoidal, Parabolic, S-Curve acceleration & deceleration
Dedicated I/O	Optically-isolated 5 - 24V with termination resistors
Dedicated Inputs (per axis)	Forward Limit (POS), Reverse Limit (NEG), Home, Amp-Fault
Dedicated Outputs (per axis)	Amp-Enable In-Position (not optically-isolated)
User I/O	24-lines of Bi-Directional User I/O Opto-isolated 5 - 24V 10 mA source or sink
Analog Inputs	8 Channels @ 12-bit resolution Configurable for 4-channel differential mode 75 kHz sampling rate 5V Unipolar input, ±2.5V Bipolar input Direct access from Host Processor
Power Requirements	<u>*4 axis*</u> +5V Icc = .6A max +12V Icc = 4mA max -12V Icc = 14mA max
Enviromental Conditions	0 - 60 degrees C 32 - 140 degrees F 20 - 95% RH, non-condensing
Construction	Full SMT; 4-layer PCB 100% bed of nails and fully functionally tested with 24-hour burn-in

STD

Interface	(STD-32/STD-80)-compatible Switch-selectable base address, I/O mapped Switch-selectable interrupts
Servo Loop Update Rate	10.0 kHz (1 axis) 3.0 kHz (4-axes simultaneously, maximum) 1.6 kHz (8-axes simultaneously, maximum) 1.25 kHz (default) User-programmable
Servo Output	±10V DC @ 16-bit resolution (from 18-bit conversion) ±18 mA current 100 ppm long-term velocity accuracy
Step Output	Maximum Step Frequency: 325 khz RS-422 line driver outputs, ±20 mA current 50% Duty Cycle Non-linearity < 1% at Full Scale
Ranges	Position: 32-bit, ±2.15 billion counts (steps) Velocity: 48-bit (±65 million counts/sec and 2 kHz sampling) Acceleration: 48-bit (±131 billion counts/sec^2 at 2 kHz sampling) Jerk: 48-bit (262 trillion counts/sec^3 at 2 kHz sampling)
Position Feedback	Input Frequency: 5 MHz (max) including Quadrature Quadrature, single-ended or differential (A,B,I) Digital Noise Filtering RS-422 line receiver inputs 4.0 mA max current input
Motion Profiles	Trapezoidal, Parabolic, S-Curve acceleration & deceleration
Dedicated I/O	TTL-compatible, 4.0 mA drive on outputs No pull-up resistors are included
Dedicated Inputs (per axis)	Forward Limit (POS), Reverse Limit (NEG), Home, Amp-Fault
Dedicated Outputs (per axis)	In-Position Amp-Enable
User I/O	2/4 axis models = 44 lines , 6/8 axis models = 24 lines TTL compatible, 4.0 mA drive on inputs Direct access from Host PC
Analog Inputs	8 Channels @ 12-bit resolution Configurable for 4-channel differential mode 75 kHz sampling rate 5V Unipolar input, ±2.5V Bipolar input Direct access from Host processor
Power Requirements	*8 axis* *4 axis* +5V Icc = .7 A max .6A max +12V Icc = 8mA max 4mA max -12V Icc = 18mA max 14Amax
Environmental Conditions	0 - 60 degrees C 32 -140 degrees F 20 - 95% RH, non-condensing
Construction	Full SMT; 4-layer PCB 100% bed of nails and fully functionally tested with 24-hour burn-in

SERCOS/STD

Interface	(STD-32/STD-80)-compatible Swtich-selectable base address I/O mapped
Fiber-Optic Connections	SMA-type Connector 1 mm plastic optical fiber Maximum length; 20 meters
Drive and I/O Interface	SERCOS (IEC 1491) compliant 1 to 8 axes Synchronous network Ring topology SERCOS loop operation: master only
Transmission Rate	2 or 4 Mbits/sec Software configurable
Block Transfer Rate	Typical: 500 Hz Maximum: 16 kHz
Drive Interpolation Rate	4x SERCOS update rate Typical: 2 kHz
Trajectory Calculation Rate	Course interpolation: 2 msec (500 Hz)
Update Rates	Position loop update rate (in drive): typical 4 kHz Velocity loop update rate (in drive): typical 5 kHz Current loop update rate (in drive): up to 20 kHz
Interoperability	Indramat, Pacific Scientific, Kollmorgen, Sanyo Denki, Modicon, and Lutze
LEDs	Axis Status/Fault Loop Closed
Motion Profiles	Trapezoidal, Parabolic, S-Curve acceleration & deceleration
Ranges	Position: 32-bit, ±2.15 billion counts (steps) Velocity: 48-bit (±65 million counts/sec and 2 kHz sampling) Acceleration: 48-bit (±131 billion counts/sec^2 at 2 kHz sampling) Jerk: 48-bit (262 trillion counts/sec^3 at 2 kHz sampling)
Power Requirements	*8 axis* *4 axis* +5V Icc = .9 A max .6A max +12V Icc = 10 mA max 4mA max -12V Icc = 20mA max 14mA max
Environmental Conditions	0 - 50 degrees C 20 - 95% RH, non-condensing
Construction	Full SMT; 4-layer PCB 100% bed of nails and fully functionally tested with 24-hour burn-in

V6U

Interface	VME compatible Switch-selectable base address, I/O mapped Switch-selectable interrupts and levels
Servo Loop Update Rate	10.0 kHz (1 axis) 3.0 kHz (4-axes simultaneously, maximum) 1.6 kHz (8-axes simultaneously, maximum) 1.25 kHz (default) User-programmable
Servo Output	±10V DC @ 16-bit resolution (from 18-bit conversion) ±18 mA current 100 ppm long-term velocity accuracy
Step Output	Maximum Step Frequency: 325 kHz RS-422 line driver outputs, ± 20 mA current 50% Duty Cycle Non-linearity < 1% at Full Scale
Ranges	Position: 32-bit, ±2.15 billion counts (steps) Velocity: 48-bit (±65 million counts/sec and 2 kHz sampling) Acceleration: 48-bit (±131 billion counts/sec^2 at 2 kHz sampling) Jerk: 48-bit (262 trillion counts/sec^3 at 2 kHz sampling)
Position Feedback	Input Frequency: 5 MHz (max) including Quadrature Quadrature, single-ended or differential (A,B,I) Digital Noise Filtering RS-422 line receiver inputs 4.0 mA max current input
Motion Profiles	Trapezoidal, Parabolic, S-Curve acceleration & deceleration
Dedicated I/O	TTL-compatible, 4.0 mA drive on outputs No pull-up resistors are included
Dedicated Inputs (per axis)	Forward Limit (POS), Reverse Limit (NEG), Home, Amp-Fault
Dedicated Outputs (per axis)	In-Position Amp-Enable
User I/O	2/4 axis models = 44 lines, 6/8 axis models = 24 lines TTL-compatible, 4.0 mA drive on outputs Direct access from Host PC
Analog Inputs	8 Channels @ 12-bit resolution Configurable for 4-channel differential mode 75 kHz sampling rate 5V Unipolar input, ±2.5V Bipolar input
Power Requirements	8 axis / 4 axis +5V Icc = .7 A max / .6A max +12V Icc = 8 mA max / 4mA max -12V Icc = 18mA max / 14mA max
Enviromental Conditions	0 - 60 degrees C 32 - 140 degrees F 20 - 95% RH, non-condensing
Construction	Full SMT; 4-layer PCB 100% bed of nails and fully functionally tested with 24-hour burn-in

104

Interface	PC-104-compatible Switch-selectable base address, I/O mapped Switch-selectable interrupts
Servo Loop Update Rate	10.0 kHz (1 axis) 3.0 kHz (4-axes simultaneously, maximum) 1.25 kHz (default) User-programmable
Servo Output	±10V DC @ 16-bit resolution (from 18-bit conversion) ±18 mA current 100 ppm long-term velocity accuracy
Step Output	Maximum Step Frequency: 325 kHz RS-422 line driver outputs, ±20 mA current 50% Duty Cycle Non-linearity < 1% at Full Scale
Ranges	Position: 32-bit, ±2.15 billion counts (steps) Velocity: 48-bit (±65 million counts/sec and 2 kHz sampling) Acceleration: 48-bit (±131 billion counts/sec^2 at 2 kHz sampling) Jerk: 48-bit (262 trillion counts/sec^3 at 2 kHz sampling)
Position Feedback	Input Frequency: 5 MHz (max) including Quadrature Quadrature, single-ended or differential (A,B,I) Digital Noise Filtering RS-422 line receiver inputs 4.0 mA max current input
Motion Profiles	Trapezoidal, Parabolic, S-Curve acceleration & deceleration
Dedicated I/O	TTL-compatible, 4.0 mA drive on outputs No pull-up resistors are included.
Dedicated Inputs (per axis)	Forward Limit (POS), Reverse Limit (NEG), Home, Amp-Fault
Dedicated Outputs (per axis)	In-Position Amp-Enable
User I/O	2/4 axis models = 20 lines TTL-compatible, 4.0 mA drive on outputs Direct access from Host PC
Power Requirements	+5V Icc = .6A max +12V Icc = 4mA max -12V Icc = 14mA max
Environmental Conditions	0 - 60 degrees C 32 - 140 degrees F 20 - 95% RH, non-condensing
Construction	Full SMT; 4-layer PCB 100% bed of nails and fully functionally tested with 24-hour burn-in

104X

Interface	PC-104-compatible Switch-selectable base address, I/O mapped Switch-selectable interrupts
Servo Loop Update Rate	10.0 kHz (1 axis) 3.0 kHz (4-axes simultaneously, maximum) 1.6 kHz (8-axes simultaneously, maximum) 1.25 kHz (default) User-programmable
Servo Output	±10V DC @ 16-bit resolution (from 18-bit conversion) ±18 mA current 100 ppm long-term velocity accuracy
Step Output	Maximum Step Frequency: 325 kHz RS-422 line driver outputs, ±20 mA current 50% Data Cycle Non-linearity < 1% at Full Scale
Ranges	Position: 32-bit, ±2.15 billion counts (steps) Velocity: 48-bit (±65 million counts/sec and 2 kHz sampling) Acceleration: 48-bit (±131 billion counts/sec^2 at 2 kHz sampling) Jerk: 48-bit (262 trillion counts/sec^3 at 2 kHz sampling)
Position Feedback	Input Frequency: 5 MHz (max) including Quadrature Quadrature, single-ended or differential (A,B,I) Digital Noise Filtering RS-422 line receiver inputs 4.0 mA max current input
Motion Profiles	Trapezoidal, Parabolic, S-Curve acceleration & deceleration
Dedicated I/O	TTL-compatible, 4.0 mA drive on outputs No pull-up resistors are included
Dedicated Inputs (per axis)	Forward Limit (POS), Reverse Limit (NEG), Home, Amp-Fault
Dedicated Outputs (per axis)	In-Position Amp-Enable
User I/O	2/4 axis models = 44 lines, 6/8 axis models = 24 lines TTL-compatible, 4.0 mA drive on outputs Direct access from Host PC
Analog Inputs	8 Channels @ 12-bit resolution Configurable for 4-channel differential mode 75 kHz sampling rate 5V Unipolar input, ±2.5V Bipolar input Direct access from Host Processor
Power Requirements	*8 axis* *4 axis* +5V Icc = .7 A max .6A max +12V Icc = 8mA max 4mA max -12V Icc = 18mA max 14mA max
Enviromental Conditions	0 - 60 degrees C 32 - 140 degrees F 20 - 95% RH, non-condensing
Construction	Full SMT; 4-layer PCB 100% bed of nails and fully functionally tested with 24-hour burn-in

SERCOS/104

Interface	PC-104-compatible Switch-selectable base address, I/O mapped
Fiber-Optic Connections	SMA type connector 1 mm plastic optical fiber Maximum length: 20 meters
Drive and I/O Interface	SERCOS (IEC 1491) compliant 1 to 8 axes Synchronous network Ring topology SERCOS loop operation; master only
Transmission Rate	2 or 4 Mbits/sec Software configurable
Block Transfer Rate	Typical: 500 Hz Maximum: 16 kHz
Drive Interpolation Rate	4x SERCOS update rate Typical: 2 kHz
Trajectory Calculation Rate	Course interpolation: 2 msec (500 Hz)
Update Rates	Position loop update rate (in drive): typical 4 kHz Velocity loop update rate (in drive): typical 5 kHz Current loop update rate (in drive): up to 20 kHz
LEDs	Axis Status/Fault Loop closed
Interoperability	Indramat, Pacific Scientic, Kollmorgen, Sanyo Denki, Modicon, and Lutze
Motion Profiles	Trapezoidal, Parabolic, S-Curve acceleration & deceleration
Ranges	Position: 32-bit, ±2.15 billion counts (steps) Velocity: 48-bit (±65 million counts/sec and 2 kHz sampling) Acceleration: 48-bit (±131 billion counts/sec^2 at 2kHz sampling) Jerk: 48-bit (262 trillion counts/sec^3 at 2 kHz sampling)
Power Requirements	*8 axis* / *4 axis* +5V Icc = .9 A max .6A max +12V Icc = 10 mA max 4mA max -12V Icc = 20 mA max 14mA max
Environmental Conditions	0 - 50 degrees C 20 - 95% RH, non-condensing
Construction	Full SMT; 4-layer PCB 100% bed of nails and fully functionally tested with 24-hour burn-in

LC

Interface	PC/XT/AT-compatible (16-bit slot) Switch-selectable base address, I/O mapped Switch-selectable interrupts
Digital Sampling Rate	10.0 kHz (1 axis) 3.0 kHz (4-axes simultaneously, maximum) 1.25 kHz (default) User programmable
Servo Output	±10V DC @ 16-bit resolution (from 18-bit conversion)
Step Outpur	Maximum Step Frequency: 325 kHz 50% Duty Cycle Non-linearity < 1% at Full Scale
Ranges	Position: 32-bit, ±2.15 billion counts (steps) Velocity: 48-bit (±65 million counts/sec and 2 kHz sampling) Acceleration: 48-bit(±131 billion counts/sec^2 at 2 kHz sampling) Jerk: 48-bit (262 trillion counts/sec^3 at 2 kHz sampling)
Position Feedback	Input Frequency: 5 MHz (max) including Quadrature Quadrature, single-ended or differential (A,B,I) Digital Noise Filtering RS-422 line receiver inputs 4.0 mA max current input
Motion Profiles	Trapezoidal, Parabolic, S-Curve acceleration & deceleration
Dedicated I/O	TTL-compatible, 4.0 mA drive on outputs No pull-up resistors are included
Dedicated Inputs (per axis)	Forward Limit (POS) Reverse Limit (NEG) Home Amp-Fault
Dedicated Outputs (per axis)	In-Position Amp-Enable
User I/O	2/4 axis models = 20 lines TTL-compatible, 4.0 mA drive on outputs Direct access from Host PC
Power Requirements	+5V Icc = .6A max +12V Icc = 4mA max -12V Icc = 14mA max
Environmental Conditions	0 - 60 degrees C 32 - 140 degrees F 20 - 95% RH, non-condensing
Construction	Full SMT; 4-layer PCB 100% bed of nails and fully functionally tested with 24-hour burn-in

SERCOS/DSP

Interface	ISA-compatible Switch-selectable base address, I/O mapped
Fiber Optic Connections	SMA type connector 1 mm plastic optical fiber Maximum length: 20 meters
Drive and I/O Interface	SERCOS (IEC 1491) compliant 1 to 8 axes Synchronous network Ring topology SERCOS loop operation: master only
Transmission Rate	2 or 4 Mbits/sec Software configurble
Block Transfer Rate	Typical: 500 Hz Maximum: 16 kHz
Drive Interpolation Rate	4x SERCOS update rate Typical: 2 kHz
Trajectory Calculation Rate	Course interpolation: 2 msec (500 Hz)
Update Rates	Position loop update rate (in drive): typical 4 kHz Velocity loop update rate (in drive): typical 5 kHz Current loop update rate (in drive): up to 20 kHz
LEDs	Axis Status/Fault Loop closed
Interoperability	Indramat, Pacific Scientific, Kollmorgen, Sanyo Denki, Modicon, and Lutze
Motion Profiles	Trapezoidal, Parabolic, S-Curve acceleration & deceleration
Ranges	Position: 32-bit, ±2.15 billion counts (steps) Velocity: 48-bit (±65 million counts/sec and 2 kHz sampling) Acceleration: 48-bit (±131 billion counts/sec^2 at 2 kHz sampling) Jerk: 48-bit (262 trillion counts/sec^3 at 2 kHz sampling)
Power Requirements	*8 axis* / *4 axis* +5V Icc = .9 A max / .6A max +12V Icc = 10 mA max / 4mA max -12V Icc = 20mA max / 14mA max
Environmental Conditions	0 - 50 degrees C 20 - 95% RH, non-condensing
Construction	Full SMT; 4-layer PCB 100% bed of nails and fully functionally tested with 24-hour burn-in

LED Support

The controller's have LEDs to indicate the status of the controller and the axes. There is one LED for each FPGA (one per four axes) and is labled 'OK'. The FPGA is a programmable component that handles the on-board logic for encoders, step and direction outputs, etc. All versions of the EPROMs and firmware support the FPGA LED:

FPGA LED	***Status***
Red	FPGA did not boot properly
No LED	DSP did not boot properly
Green	FPGA and DSP are OK

There is one LED for each axis and is labled 0, 1, 2, 3, 4, 5, 6, or 7. EPROM versions 1.24, 2.24 and higher and firmware versions 2.1C and higher support the LEDs. Older firmware (version 2.1C) does not support the axis LEDs. When using 2.1C firmware the axis LEDs may remain Orange or may not be lit:

Axis LED	***Status***
No LED/Off	Axis not enabled
Orange	Reset in Progress
Red	Idle Mode (generated by an ABORT_EVENT)
Green	Run Mode
Flashing Green	Command Velocity is Non-Zero

APPENDIX F OPTOCON REFERENCE

The Optical Isolation Connection Module (OptoCon) is a connection accessory for Motion Engineering's LC/DSP and 104/DSP motion controllers. The OptoCon converts a 50-pin ribbon cable (from the motion controller) to screw terminal connections. The OptoCon replaces the standard passive Phoenix Contact terminal block (STC-50) with an *active* terminal block that provides optical isolation and fused overvoltage protection for dedicated and user I/O.

The OptoCon and STC-50 have the same physical dimensions. The pinouts are identical except that a ground and +5 volt connection on the screw terminal block have been replaced with an opto-ground and an opto-Vcc (5-24 volts). You use 2 microswitches configure the direction of 3 user I/O ports.

Each OptoCon supports 2 motion control axes, dedicated I/O (2 axes) and 10 lines of user I/O. Connector P1 is a 50-pin IDC connector, and is compatible with the LC/DSP and 104/DSP controllers. Four-axis applications require using 2 OptoCon modules. The OptoCon requires that you use the CBL-100 cable, with each CBL-100 cable supporting up to 4 axes of control.

Switch Settings

Switch S1

The dedicated output circuits (Amp Enable & In Position) of the OptoCon have pull-down resistors on their inputs that prevent unwanted output transitions during a motion controller reset or power-up sequence. Refer to the *Output Circuit* figure on page F-8. To disable the pull-down resistors, use switch S1.

Note	The **Amp Enable pull-down resistors should only be enabled** when the Amp Enables are configured as **Active High** on the motion controller.

If either of the Amp Enable outputs are configured as *Active Low*, the appropriate pull-down resistor should be disabled (as indicated in the next table). To configure the Amp Enables for *Active High* or *Active Low* operation, use the MEI library function **set_boot_amp_enable_level(...)**.

Note	**The pull-down resistors for the *In Position* outputs should always be enabled**, because the the *In Position* outputs are always *Active High*.

Table F-1 Switch S1 Settings (To enable/disable pull-down resistors)

	Position	*Setting*	*Signal*	*Pull-Down Resistor is*
Switch S1	1	On	Amp Enable(0/2)	Enabled
	1	Off	Amp Enable(0/2)	Disabled
	2	On	In Position(0/2)	Enabled
	2	Off	In Position(0/2)	Disabled
	3	On	Amp Enable(1/3)	Enabled
	3	Off	Amp Enable(1/3)	Disabled
	4	On	In Position(1/3)	Enabled
	4	Off	In Position(1/3)	Disabled

Switches S2, S3

To configure the User I/O opto-isolation circuitry as inputs or outputs, use switches S2 and S3. To set the input and output directions, use the settings in the next table.

The directions set with the switches should match those set on the controller using the MEI library function **init_boot_io(...)**, so that the OptoCon and the DSP controller are configured identically at power-up. After using **init_boot_io(...)** to configure a port's direction, do not use **init_io(...)** to reconfigure the port's direction.

Warning!	**You can only use the switch settings shown in the table.** Other switch settings may damage the OptoCon circuits.

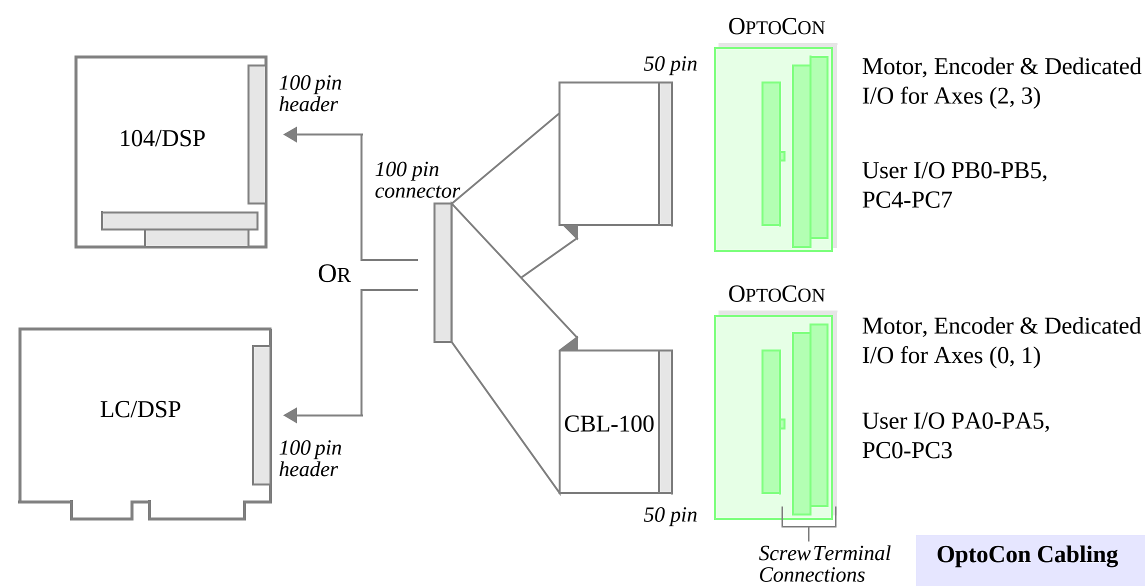
Table F-2 Switch S2/S3 Settings (To configure User I/O as inputs or outputs)

	Position	*Port A/B Input Port C Input*	*Port A/B Input Port C Output*	*Port A/B Output Port C Input*	*Port A/B Output Port C Output*
Switch S2	1	*Off*	*Off*	On	On
	2	*Off*	On	*Off*	On
	3	*Off*	*Off*	On	On
	4	*Off*	*Off*	On	On
Switch S3	1	*Off*	*Off*	On	On
	2	*Off*	*Off*	On	On
	3	*Off*	*Off*	On	On
	4	*Off*	*Off*	On	On
	5	*Off*	On	*Off*	On
	6	*Off*	On	*Off*	On
	7	*Off*	On	*Off*	On
	8	*Off*	On	*Off*	On

Installation

Before connecting any cables or wires to the OptoCon, you must correctly set the switches as described in the preceding section. **Only the switch settings shown in the table are allowed!** Other switch settings may cause damage to the OptoCon module and the DSP controller.

Connect the 100-pin connector on MEI accessory cable CBL-100 to the 100-pin header on the LC/DSP or 104/DSP. Connect either of the two 50-pin connectors on the CBL-100 to the 50-pin header on the OptoCon.



OptoCon Cabling

Screw Terminal Connectors

For Axes 0, 1

Table F-3 Screw Terminal Connector (Axes 0, 1)

Pin	Signal	Pin	Signal
1	+5V	2	V_USER
3	Encoder A(0) +	4	Encoder A(1) +
5	Encoder A(0) -	6	Encoder A(1) -
7	Encoder B(0) +	8	Encoder B(1) +
9	Encoder B(0) -	10	Encoder B(1) -
11	Encoder Index(0) +	12	Encoder Index(1) +
13	Encoder Index(0) -	14	Encoder Index(1) -
15	+/- 10V Analog Out(0)	16	+/- 10V Analog Out(1)
17	GND	18	USER_GND
19	Step Pulse(0) +	20	Step Pulse(1) +
21	Step Pulse(0) -	22	Step Pulse(1) -
23	Direction(0) +	24	Direction(1) +
25	Direction(0) -	26	Direction(1) -
27	Positive Limit(0)	28	Positive Limit(1)
29	Negative Limit(0)	30	Negative Limit(1)
31	Home Input(0)	32	Home Input(1)
33	Amp Fault(0)	34	Amp Fault(1)
35	Amp Enable(0)	36	Amp Enable(1)
37	In Position(0)	38	In Position(1)
39	User I/O PA0	40	User I/O PA3
41	User I/O PA1	42	User I/O PA4
43	User I/O PA2	44	User I/O PA5
45	User I/O PC0	46	User I/O PC2
47	User I/O PC1	48	User I/O PC3
49	GND	50	USER_GND

Shaded signals are optically isolated.

For Axes 2, 3

Table F-4 Screw Terminal Connector (Axes 2, 3)

Pin	Signal	Pin	Signal
1	+5V	2	V_USER
3	Encoder A(2) +	4	Encoder A(3) +
5	Encoder A(2) -	6	Encoder A(3) -
7	Encoder B(2) +	8	Encoder B(3) +
9	Encoder B(2) -	10	Encoder B(3) -
11	Encoder Index(2) +	12	Encoder Index(3) +
13	Encoder Index(2) -	14	Encoder Index(3) -
15	+/- 10V Analog Out(2)	16	+/- 10V Analog Out(3)
17	GND	18	USER_GND
19	Step Pulse(2) +	20	Step Pulse(3) +
21	Step Pulse(2) -	22	Step Pulse(3) -
23	Direction(2) +	24	Direction(3) +
25	Direction(2) -	26	Direction(3) -
27	Positive Limit(2)	28	Positive Limit(3)
29	Negative Limit(2)	30	Negative Limit(3)
31	Home Input(2)	32	Home Input(3)
33	Amp Fault(2)	34	Amp Fault(3)
35	Amp Enable(2)	36	Amp Enable(3)
37	In Position(2)	38	In Position(3)
39	User I/O PB0	40	User I/O PB3
41	User I/O PB1	42	User I/O PB4
43	User I/O PB2	44	User I/O PB5
45	User I/O PC4	46	User I/O PC6 (or DSP Interrupt)
47	User I/O PC5	48	User I/O PC7 (or PC Interrupt)
49	GND	50	USER_GND

Shaded signals are optically isolated.

Specifications

All optically isolated outputs (Amp Enables, In Position bits, User I/O) and the V_USER input are protected by automatic fuses. When tripped, these fuses automatically reset themselves within a few seconds.

Operating temperature range	0 – 50° C
Isolation voltage	2500 V_{RMS}
V_USER voltage range	5 – 24 V_{DC}
V_USER voltage fuse trip current	1 A

Table F-5 Inputs

	V_USER = 5 VDC	***V_USER = 24 VDC***
"On" threshold voltage	0.6 V max	19 V max
Propagation delay High-Low, t_{PDHL}	50 µsec max	20 µsec max
Propagation delay Low-High, t_{PDLH}	300 µsec max	400 µsec max

Table F-6 Outputs

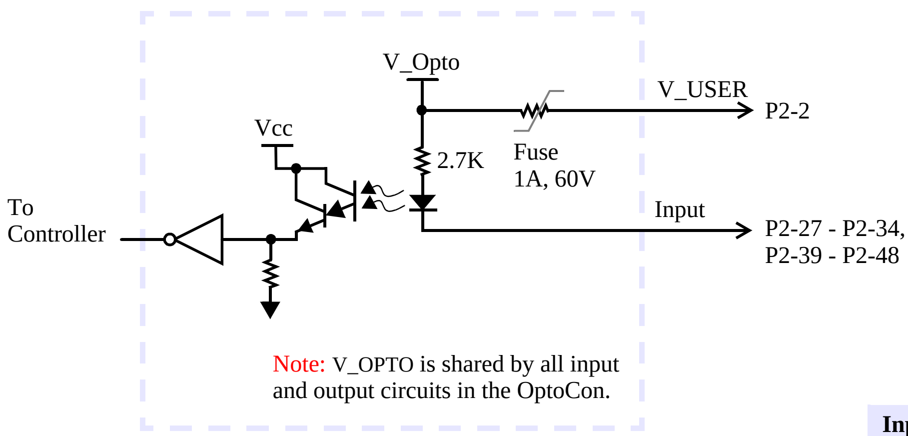
	V_USER = 5 VDC	***V_USER = 24 VDC***
"On" state output *voltage*	0.25 V @ 250mA	0.25 V @ 250mA
"On" state output *current*	250 mA max	250 mA max
"Off" state output *leakage current*	25 µA max	25 µA max
Propagation delay Low-High, t_{PDLH}	10 µsec max	20 µsec max
Propagation delay High-Low, t_{PDHL}	300 µsec max	100 µsec max
Output *rise time*, t_R	5 µsec max	5 µsec max
Output *fall time*, t_F	75 µsec max	25 µsec max

Schematics

All OptoCon input and output circuits are electrically identical. To program the User I/O signals (OptoCon 1: PA0-5, PC0-3; OptoCon 2: PB0-5, PC4-7) as inputs or outputs, use the switches S2 and S3 on the OptoCon and in conjunction with the MEI library function **init_boot_io(...)** on the motion controller. After using **init_boot_io(...)** to configure a port's direction, do not use **init_io(...)** to reconfigure the port's direction.

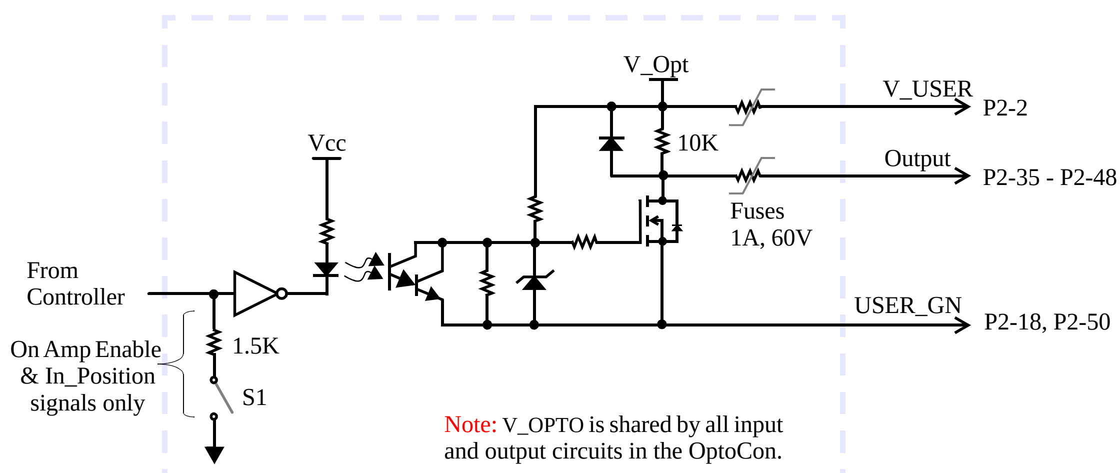
The Dedicated I/O signals (Amp Enable, In Position, Positive Limit, Negative Limit, Home and Fault) **cannot be reconfigured**. All of the I/O signals share a common supply, (V_USER/USER_GND), which is fused at 1 amp. Additionally, each individual output is fused at 1 amp.

TYPICAL INPUT CIRCUIT



Input Circuit

TYPICAL OUTPUT CIRCUIT



Output Circuit

Circuit Examples

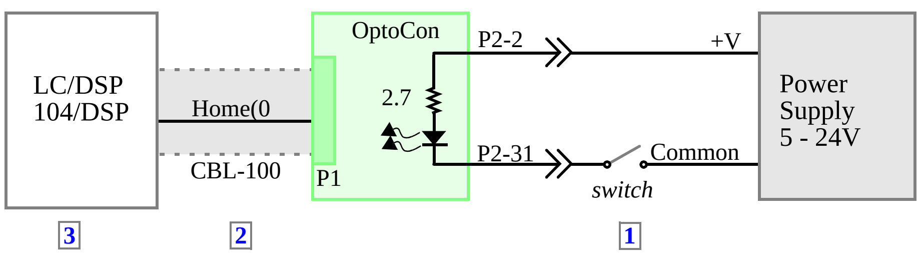
Connect an OptoCon Input to a Switch

The next figure shows how to connect an OptoCon input to detect the state of a Home switch. This circuit will also work for *any* of the OptoCon inputs.

Use the MEI library functions **set_home_level(...)** or **set_boot_home_level(...)** to configure the Home(0) input on the MEI motion controller for either *Active High* or *Active Low* event generation logic.

The truth table shows the values that the motion controller will read, depending upon the state of the switch and the configuration of the Home event logic. For example, if the switch is open, the Home input will be high (1), and if the Home event logic is configured for *Active High*, the controller will generate an event.

For input circuitry, see schematics on page F-8.



1 Switch State	2 Home Input (State)	3 Active High (Event?)	Active Low (Event?)
Open	1	Yes	No
Closed	0	No	Yes

*Use **home_switch(...)** to read the Home input state.*

*Use **axis_state(...)** to read the event generation.*

OptoCon Input to Switch

Connect an OptoCon Input to an Open Collector Driver

The next figure shows how to connect an OptoCon input to detect the state of an open collector driver. This circuit will also work for *any* of the OptoCon inputs.

Use the MEI library functions **set_home_level(...)** or **set_boot_home_level(...)** to configure the Home(0) input on the MEI motion controller for either *Active High* or *Active Low* event generation logic.

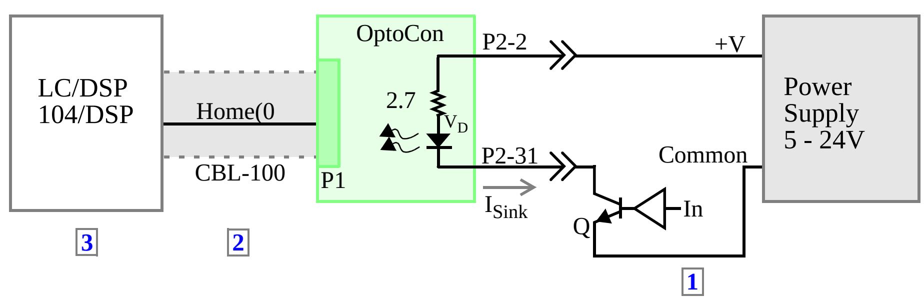
The truth table shows the values that the motion controller will read, depending upon the state of the driver transistor and the configuration of the Home event logic. For example, if the *In* = 1 (turning the transistor *On*), the Home input will be low (0), and if the Home event logic is configured for *Active High*, the controller will *not* generate an event.

When *In* is high, the driver transistor is required to sink the current flowing through the opto-isolator diode. The driver transistor must be capable of sinking this current. To calculate I_C:

$I_{Sink} \cong (V - V_D - V_{CE}) / 2700$
V = Your system's power supply voltage
V_D = Voltage across diode, $V_D \cong 1V$
V_{CE} = Collector-emitter "*On*" *voltage* for Q

For V = 24V, V_{CE} = 0.2V and $I_C \cong$ 8.4 mA.
For V = 5V, V_{CE} = 0.2V and $I_C \cong$ 1.4 mA.

For input circuitry, see schematics on page F-8.



1 2 3

In	Transistor "Q"	Home Input (State)	Active High (Event?)	Active Low (Event?)
1	On	0	No	Yes
0	Off	1	Yes	No

*Use **home_switch(...)** to read the Home input state.*

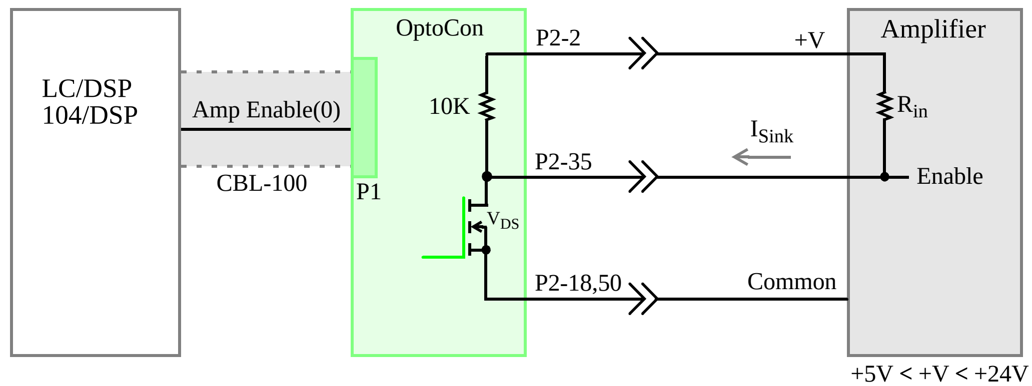
*Use **axis_state(...)** to read the event generation.*

Open Collector Driver to OptoCon Input

Connect an OptoCon Output to an Amplifier Enable Input

Using an Internal Pull-Up Resistor

In the next figure, the Enable input on the amplifier has an internal pull-up resistor (R_{in}). You can use this configuration for either *Active High* or *Active Low* Amp Enable inputs.



For output circuitry, see schematics on page F-8.

OptoCon Output to Amp Enable (Pull-Up Resistor)

Use Motion Console's *Axis Configuration* under the *Axis Operation* window to configure the Amp Enable output on the MEI motion controller for either *Active High* or *Active Low* detection.

Note	The Amp Enable output's **polarity must match** the polarity of the amplifier's Enable input. The Amp Enable output and the amplifier's Enable input must be either both *Active High* or both Active Low.

In order for the OptoCon to work correctly in this configuration, I_{sink} must be less than the maximum *"On" state* output current for the OptoCon (250 mA), otherwise the OptoCon may not be able to disable the amplifier by pulling the Enable input low. To calculate I_{sink}:

$I_{sink} \cong (V - V_{DS}) / R_P$

R_P = Equivalent parallel resistance of R_{in} & 10K, $R_P = R_{in} * 10K / (R_{in} + 10K)$

V = Amplifier logic *power supply voltage*

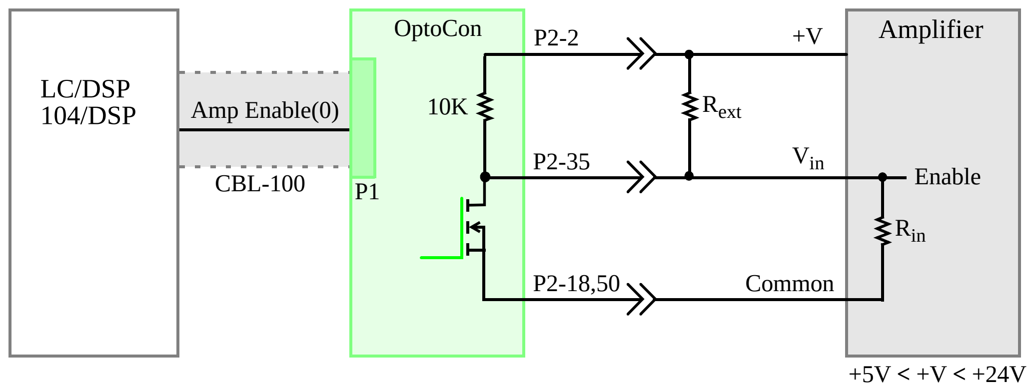
V_{DS} = OptoCon "On" state *output voltage*, $V_{DS} < 0.25V$

R_{in} = Amplifier Enable internal pull-up resistance

Warning!	**You must set S1 correctly for "Active High" or "Active Low" Amp Enable Operation.** (see *Switch Settings* on page F-2)

Using an Internal Pull-Down Resistor

The next figure shows how to connect the OptoCon to an amplifier's Enable input that has a pull-down resistor (that is inside the amplifier). This configuration can be used for either *Active High* or *Active Low* amplifier Enable inputs.



For output circuitry, see schematics on page F-8.

OptoCon Output to Amp Enable (Pull-Down Resistor)

Use the MEI library function **set_amp_enable_level(...)** or **set_boot_amp_enable_level(...)** to configure the Amp Enable output on the MEI motion controller for either *Active High* or *Active Low* detection.

Note:	The Amp Enable output's **polarity must match** the polarity of the amplifier's Enable input. The Amp Enable output and the amplifier's Enable input must be either both *Active High* or both Active Low.

In order for the OptoCon to work correctly in this configuration, V_{in} must exceed the amplifier manufacturer's minimum "high" input *threshold voltage*. The "high" level at V_{in} is determined by the voltage divider between the OptoCon pull-up resistor (10K) and R_{in}. To calculate V_{in}:

$V_{in} \cong V * (R_{in} / (R_{in} + 10K))$
V = Amplifier logic *power supply voltage*
R_{in} = Amplifier Enable internal pull-up resistance

If the value for V_{in} **is lower than** the amplifier manufacturer's minimum "high" input *threshold voltage*, you must add the resistor R_{ext} (see the next figure). To calculate the required value of R_{ext}, first calculate the parallel resistance (R_P) required to achieve the desired V_{in} 'high' level.

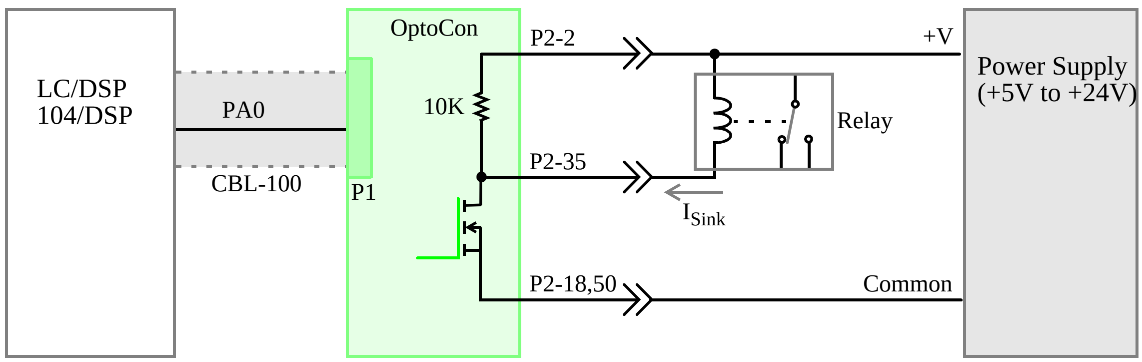
$R_P = R_{in} * (-1 + V/ V_{in})$
V = Amplifier's logic *power supply voltage*
V_{in} = Required amplifier Enable "high" *input voltage*
R_{in} = Amplifier Enable internal pull-up resistance

Next calculate R_{ext}, so that R_{ext} in parallel with 10K *is equal to* R_P.
$R_{ext} = R_P / (1 - R_P /10K)$

Warning!	**You must set S1 correctly for "Active High" or "Active Low" Amp Enable Operation.** (see *Switch Settings* on F-2)

Connect an OptoCon Output to a Relay

The next figure shows how to drive a relay using one of the User I/O (PA0) signals from the motion controller via the OptoCon. This circuit can be used with *any* of the OptoCon outputs.



For output circuitry, see schematics on page F-8.

OptoCon Output to A Relay

When PA0 is set 'low', the relay is energized. For the OptoCon to work correctly in this configuration, I_{sink} must be smaller than the maximum "On" state *output current* for the OptoCon (250 mA). If this condition is not met, the relay may not switch. To calculate I_{sink}:

$I_{sink} \cong (V - V_{DS}) / R_C$

V = Amplifier logic *power supply voltage*

V_{DS} = OptoCon "On" state *output voltage*, $V_{DS} < 0.25V$

R_C = Relay coil resistance

Connect an OptoCon Output to a Relay

INDEX

Numerics

A

B

C

D

E

F

G

H

I

K

L

M

O

P

Q

S

T

U

V

V