Prodigy®/CME Machine-Controller User's Guide

P M D

Performance Motion Devices, Inc.
80 Central Street
Boxborough, MA 01719

Revision 1.0, July 2013

NOTICE

Related Documents

Magellan® Motion Processor User's Guide

Complete description of the Magellan Motion Processor features and functions with detailed theory of its operation.

Atlas Digital Amplifier User's Manual

Description of the Atlas Digital Amplifier electrical and mechanical specifications along with a summary of its operational features.

Atlas Digital Amplifier Complete Technical Reference

Complete technical and mechanical description of the Atlas Digital Amplifier with detailed theory of operations.

Magellan® Motion Processor Programmer's Command Reference

Descriptions of all Magellan Motion Processor commands, with coding syntax and examples, listed alphabetically for quick reference.

PMD Resource Access Protocol Programmer's Reference

Descriptions of all Prodigy/CME product commands, with software architecture overview, command syntax, and examples.

C-Motion® Engine Development Tools

Description of the C-Motion Engine hardware resources, development environment, software tools, and command set.

Pro-Motion® User's Guide

User's guide to Pro-Motion, the easy-to-use motion system development tool and performance optimizer. Pro-Motion is a sophisticated, easy-to-use program which allows all motion parameters to be set and/or viewed, and allows all features to be exercised.

Table of Contents

This page intentionally left blank.

List of Figures

This page intentionally left blank.

1. Installation

In This Chapter

1.1 Prodigy Family Overview

The Prodigy® family of motion control cards provide high-performance control of DC brush, brushless DC, and step motors. They are available in four formats: PCI bus, PC/104 bus, Stand-Alone, and Machine-Controller, and allow multiple forms of communication including PCI bus, PC/104 bus, serial, CANbus, and Ethernet.

All Prodigy cards are based on PMD's Magellan® Motion Processors, which perform high speed motion control functions such as profile generation, servo loop closure, pulse & direction signal generation, and many other real-time functions. Certain versions of Prodigy cards include a C-Motion® Engine (CME), which allows the user to offload application code from the host onto the motion card. The Machine-Controller cards provide the option of on-board amplification via PMD's Atlas Amplifiers.

The following product selector table summarizes the various members of the Prodigy motion card family and associated User's Guide documentation.

Prodigy P/N	Format	C-Motion Engine	On-board Amplifier	User Guide
PR8258x20	PC/104	No	No	Prodigy-PC/104 User's Guide
PR8358x20	PC/104	Yes	No	Prodigy/CME-PC/104 User's Guide
PR9258x20	PCI	No	No	Prodigy-PCI User's Guide
PR9358x20	PCI	Yes	No	Prodigy/CME-PCI User's Guide
PR13x58x20	Stand-Alone	Yes	No	Prodigy/CME Stand-Alone User's Guide
PR33xxabcd58x20	Machine-Controller	Yes	Yes	Prodigy/CME Machine-Controller User's Guide

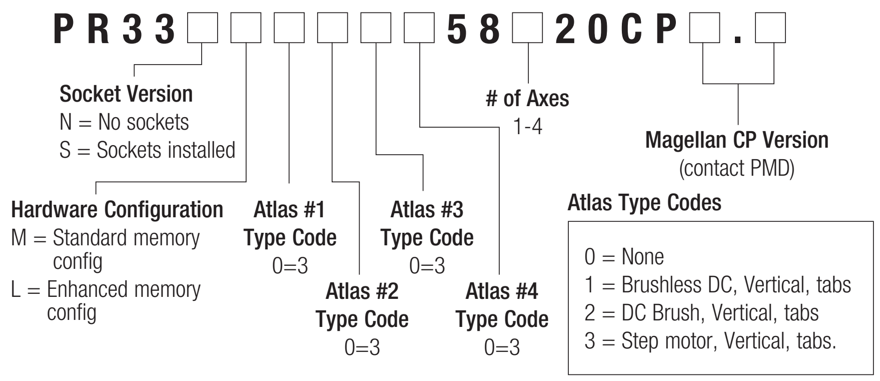
1.2 Prodigy/CME Machine-Controller

This manual provides a complete User's Guide for the Prodigy/CME Machine-Controller cards. For documentation on other members of the Prodigy family please consult the appropriate documentation.

Prodigy/CME Machine-Controller cards, also referred to as 'machine controllers' come in various configurations of axis number (from 1 to 4). In addition, they may be provided with specific pre-soldered Atlas amplifiers, with Atlas amplifier sockets, or without Atlas amplifiers.

1.2.1 Prodigy/CME Machine-Controller Part Number

Figure 1-1: Prodigy/CME Machine-Controller Part Numbering



1.2.2 Prodigy/CME Machine-Controller Part Number Examples

Part Number	Description	Atlas Units
PR33NM111158420	Four axis with standard memory configuration and Atlas amplifiers soldered onto card	Four brushless DC Atlas
PR33NL000058320	Three axis with extended memory option and no installed Atlas units	none
PR33NM320058220	Two axis with standard memory configuration and Atlas amplifiers soldered onto card	Step motor Atlas in axis #1 slot DC brush Atlas in axis #2 slot

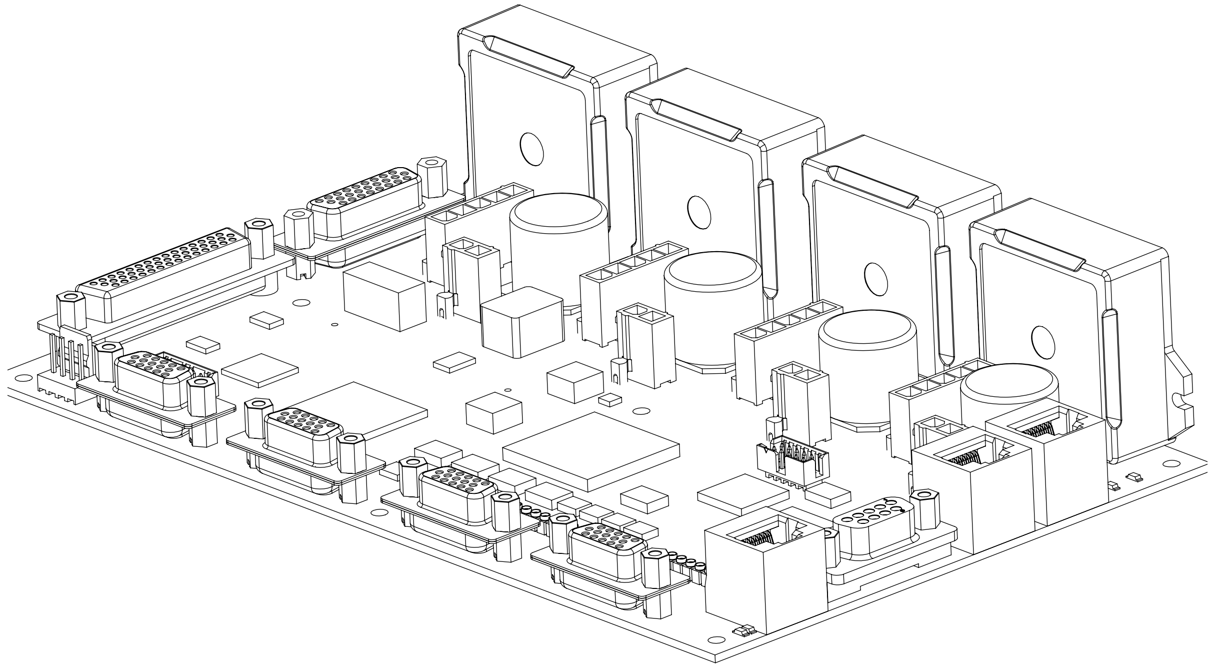
1.2.3 On-card Amplifier Socket Configuration

The Prodigy/CME Machine-Controller cards support up to four on-card Atlas amplifiers. In production use of the machine controller any installed Atlas amplifiers should be soldered to the board. However for prototyping purposes it may be desirable to utilize Atlas-compatible sockets to allow amplifiers to be installed into the card as needed. As shown in the part numbering information above, whether or not Atlas sockets are installed is specifiable when ordering the card.

If the Prodigy/CME Machine-Controller card is ordered with Atlas units specified but with sockets unspecified, then the card is provided with Atlas units soldered on.

Users who do not plan to use on-card amplification, or who plan to solder Atlas units on themselves, should specify the no socket configuration of the card.

Note that if the DK (developer's kit) version of the card is specified, sockets are always installed.



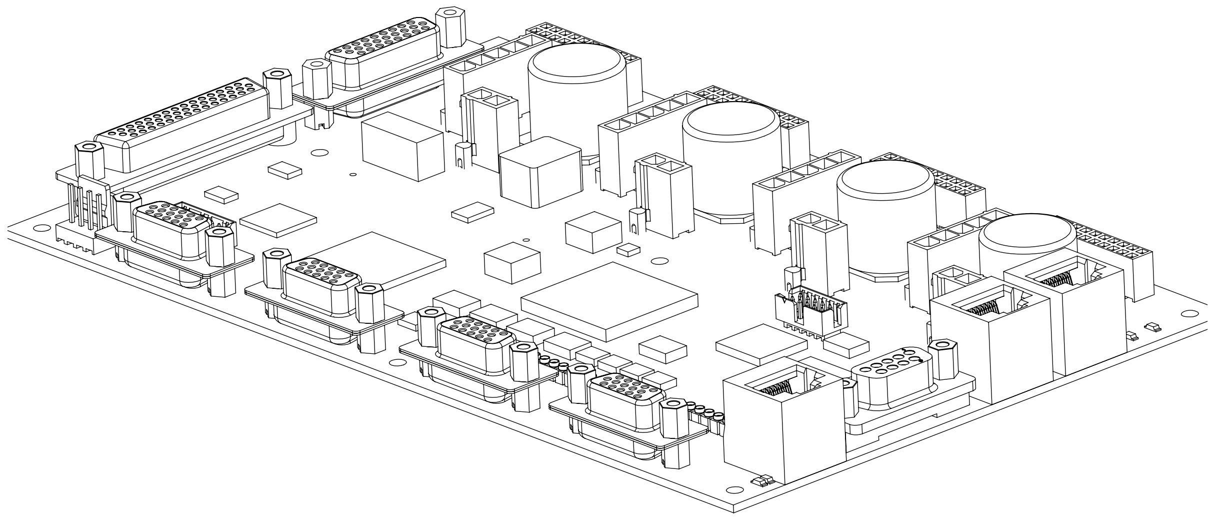


Figure 1-2: Machine-Controller Card With and Without Atlas Units

1.2.4 Hardware Configuration

The Prodigy/CME Machine-Controller cards support two different on-card memory configurations, one with standard memory installation and one with enhanced memory installation. The following table shows this:

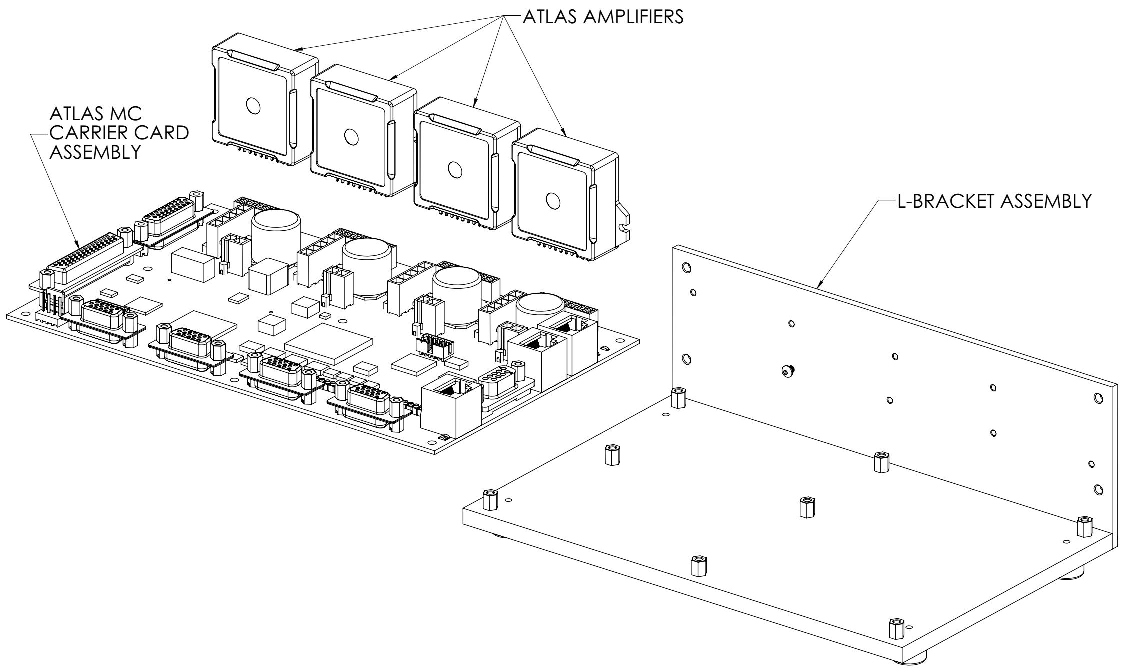
Memory Configuration	On-card dual ported RAM
Standard	128 Kbytes
Enhanced	468 Kbytes

The standard memory configuration is sufficient for most applications, however applications that require large trace storage memory may benefit from the enhanced memory option.

1.3 Machine-Controller Developer's Kit

To simplify development a Machine-Controller Developer's Kit is available. Some of the components of this developer's kit are shown in Figure 1-3.

Figure 1-3: Developer Kit Components



1.3.1 Machine-Controller Developer's Kit Part Number

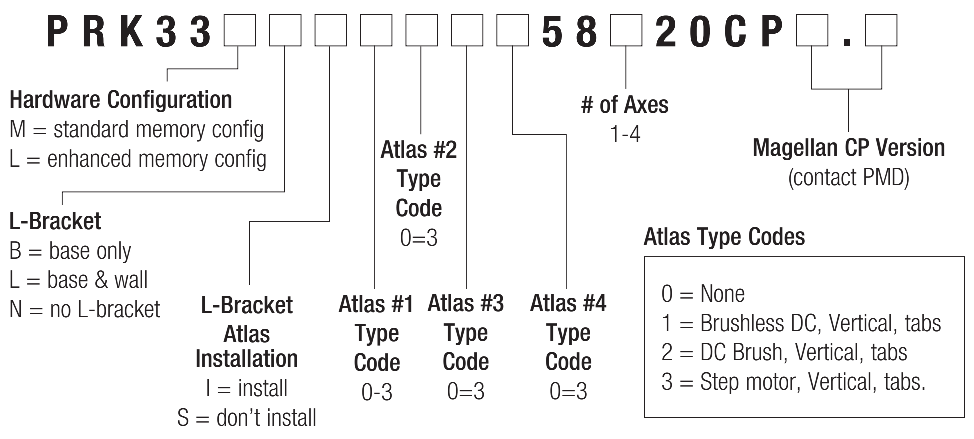


Figure 1-4: Prodigy/CME Machine-Controller DK Part Numbering

1.3.2 Developer's Kit Part Number Examples

Part Number	Description	Atlas Units
PRK33MLI111158420	Four axis DK with standard memory configuration and Atlas amplifiers mounted on full L-bracket (base & vertical plate)	Four brushless DC Atlas
PRK33LBS123058320	Three axis DK with enhanced memory option. Card is mounted on base and shipped with Atlas units	Brushless DC motor Atlas DC brush Atlas Step motor Atlas
PRK33MLS222058320	Three axis with standard memory configuration and full L-bracket and Atlas amplifiers shipped with card	Three DC brush Atlas units

1.3.3 L-Bracket Configuration

As shown in Figure 1-3 an L-bracket assembly is available with the machine controller card. This L-bracket provides a number of important functions including:

- Increases heat sinking capacity
- Provides stable base for bench-top prototyping
- Provides rigid support for vertical Atlas amplifiers
- May be helpful for mounting the card in the destination machine hardware

Most developer's kits are ordered with the L-bracket for the reasons indicated above, however there is no requirement to do so.

For complete mechanical drawings of the L-bracket component see Section 3.8, "L-Bracket Mechanical Dimensions."

1.3.4 L-Bracket Atlas Installation

When ordering the developer's kit you may choose to have the Atlas units mounted onto the L-bracket, or you may choose to have all the components shipped to you but not assembled.

Generally, having the Atlas units already installed is recommended to simplify setup, but there is no requirement for this and you can specify either configuration when you order the developer's kit.

1.3.5 Developer's Kit Components List

The Magellan Motion Processor Developer's Kit contains the following components:

- 4-axis Prodigy/CME Machine-Controller card
- Optional L-bracket hardware
- Prodigy/CME SDK CD-ROM containing C-Motion and documentation in PDF format
- Stub cable set for various card connectors
- Pro-Motion CD-ROM

Documentation:

- *Prodigy/CME Machine-Controller User's Guide*
- *Magellan Motion Processor User's Guide*
- *Magellan Motion Processor Programmer's Command Reference*
- *PMD Resource Access Protocol Programmer's Reference*
- *C-Motion Engine Development Tools*
- *Pro-Motion User's Guide*
- *Atlas Digital Amplifier User's Manual* (if DK ordered with one or more Atlas amplifiers)

If any of these components are missing, please contact your PMD representative.

1.4 Software

Four major software packages are provided with the Prodigy/CME Machine-Controller cards:

Pro-Motion®, an interactive Windows-based exerciser & software development tool

C-Motion®, aC-language library that allows you to create motion applications using the C programming language

C-Motion® Engine Development Tools, a set of development resources that allow you to create, download, and monitor programs loaded in the Prodigy/CME Machine-Controller card's C-Motion Engine

Visual Basic Support, a DLL and source code that lets you create motion applications using the Visual Basic programming language

Here is more information on each of these software packages:

1.4.1 Pro-Motion

Pro-Motion is a sophisticated, easy-to-use exerciser program which allows all card parameters to be set and/or viewed, and allows all card features to be exercised. Pro-Motion features include:

- Motion oscilloscope graphically displays motion processor parameters in real-time
- Axis Wizard to automate axis setup and configuration
- Interactive servo tuning
- Atlas Amplifier setup and configuration
- Project window for accessing motion resources and connections
- Ability to save and load current settings
- Distance and time units conversion
- Motor-specific parameter setup
- Axis shuttle performs continuous back and forth motion between two positions
- C-Motion Engine monitor/debug window
- C-Motion Engine user application code download

Pro-Motion is described in the *Pro-Motion User's Guide.*

1.4.2 C-Motion

C-Motion provides a convenient set of callable routines comprising the code required for controlling the Magellan Motion Processor and PRP-controlled resources, whether running on a separate host computer such as a PC, or running on the Prodigy/CME card in the C-Motion Engine. C-Motion includes the following features:

- Magellan axis virtualization
- Support for Atlas Digital Amplifiers
- Ability to communicate to multiple PMD motion cards or modules
- Ability to communicate via PCI bus, PC/104 bus, serial, CANbus, or Ethernet
- Provided as source code, allowing easy compilation & porting onto various run-time environments including PC, microprocessor, embedded card, or C-Motion Engine
- Can be easily linked to any C/C++ application

C-Motion is described in the *Magellan Motion Processor Programmer's Command Reference* and the *PMD Resource Access Protocol Programmer's Reference.*

1.4.3 C-Motion Engine Development Tools

The C-Motion Engine Development Tools include a source code editor, compiler, linker, and Pro-Motion-based code management & monitoring tools that allow the user to quickly and easily develop applications that will run on a device equipped with a C-Motion Engine. The C-Motion Engine Development Tools includes the following features:

- Complete toolset for creation of user-specific applications running on the motion card
- Open source compiler and C libraries
- Interactive Development Environment

- Supports PCI bus, PC/104 bus, RS232/RS485, CANbus, and Ethernet communications

The C-Motion Engine development tools are described in the *C-Motion Engine Development Tools Manual.*

1.4.4 Visual Basic Support

VB-Motion provides a complete set of methods and properties for developing applications in Visual Basic using a dynamically loaded library (DLL) containing PMD library software. The DLL may also be used from any language capable of calling C language DLL procedures, but no special software support is provided.

VB-Motion includes the following features:

- Magellan axis virtualization
- Ability to communicate to multiple PMD motion cards or Modules
- Ability to communicate via PCI bus, serial, CANbus, or Ethernet
- Provided as a single DLL and Visual Basic .NET source code for easy porting onto various PC environments

VB-Motion is documented in the *PMD Resource Access Protocol Programmer's Reference.*

1.5 Accessory Products

The Prodigy/CME Machine-Controller cards can be enhanced with the addition of any or all of the hardware accessory products described in the following table.

Component Part Number	Description
Cable-4355-01.R	Dual serial cable. This cable connects to the Prodigy/CME Machine-Controller's dual serial connector and provides two DB-9 connectors suitable for connection to a PC serial port or USB to serial converter.
Cable-4705-KIT-01.R	CANbus connector and terminator. This cable connects to the card's CANbus connector and has RJ45 connectors on both ends.
Cable-RJ45-02-R	Ethernet connector. This cable connects to the card's Ethernet connector, and has RJ45 connectors on both ends.
BIMC-0001-01	Breakout interconnect module and cable provides convenient jack-screw type terminators for the 15-pin Axis Connectors.
BIMC-0002-01	Breakout interconnect module and cable provides convenient jack-screw type terminators for the 26-pin Amplifier I/O Connector.
BIMC-0003-01	Breakout interconnect module and cable provides convenient jack-screw type terminators for the 44-pin General I/O Connector.
Adapt-USB232-01.R	This adapter provides USB to serial conversion. It is useful for connecting to the Prodigy/ CME's DB-9 serial port from a USB port.

For information on ordering these accessory products, please contact your PMD representative.

1.6 Installation Overview

1. Before using the card, the software must be installed. See Section 1.8, "Software Installation," for instructions on installing the software.

2 For a normal installation of a Prodigy/CME Machine-Controller card you will need to configure the card for the specific motion hardware to which it will be connected. See Section 1.9, “Preparing the Card for Installation” for a description of configuring the cards.

3 Next, connect the system’s motors, encoders, amplifiers, and sensors to operate the motion hardware. See Section 1.10, “Connection Summary” for a description of the available connections and options for the Prodigy/CME Machine-Controller card.

4 Connect the Prodigy/CME Machine-Controller card to the host PC via an Ethernet cable and a Serial cable. During first time setup Ethernet communications will be used, but during development you can use any of the available communication options; serial, CANbus, or Ethernet. See Section 1.10, “Connection Summary,” for a description of the communications connections and options.

5 Once this hardware configuration is complete, the final step to finish the installation is to perform a functional test of the finished system. See Section 1.12, “First-Time System Verification,” for a description of this procedure.

Once these steps have been accomplished, the installation is complete, and the card is ready for operation.

1.7 Recommended Hardware

To install a Prodigy/CME Machine-Controller card the following hardware is recommended.

- Intel (or compatible) processor, Pentium or better, 300 MB of available disk space, 256MB of available RAM, and a CD-ROM drive. The supported PC operating systems are Windows XP/Vista/Windows 7
- One to four Atlas Digital Amplifiers. The type of Atlas amplifier depends on the type of motor being used, either DC brush, brushless DC, or step motor. Depending on how you ordered your machine controller card the Atlases may already be installed, or you may need to install them yourself. Note that the machine controller card can be operated without on-card Atlas amplifiers using off-card module-based amplifiers. However, for simplicity these first-time installation instructions will assume the use of Atlas amplifiers.
- One to four step, DC brush, or brushless DC motors. These motors may or may not provide encoder position feedback signals, depending on the type of motor being used. Encoder feedback is a requirement for DC brush and brushless DC motors. For step motors, encoder feedback is an option.
- Cables as required to connect to the motors you have selected. Each motor axis that is driven by an Atlas amplifier requires a motor drive cable. In addition, feedback signals such as encoder feedback and limits are provided via a separate connector, one connector per axis.
- Power supply, power cable, and communication cables. Each Atlas receives separate power input at the desired motor voltage. The machine controller card itself is powered from the Atlas power input for axis #1 using on-card DC to DC circuitry. For first-time installation you will connect to both the card's serial port and Ethernet port.

1.8 Software Installation

Two CD-ROMs comprise the software distribution for Prodigy/CME cards. All software applications are designed to work with Windows XP, Vista, or Windows 7.

- Pro-Motion: The Pro-Motion CD-ROM is located in its own Pro-Motion box, and contains the software associated with the Pro-Motion Optimized Motion System Development software.

- PMD Prodigy/CME SDK: The PMD Prodigy/CME SDK CD-ROM is located separately, and contains the C-Motion Engine development tools, the C-Motion source libraries, and Visual Basic Libraries.

To install the software:

1. Insert the Pro-Motion disk into the CD-ROM drive of your computer.
   - If autorun is enabled, the installation process will begin when the CD-ROM is inserted.
   - If autorun is not enabled, go to the next step.
2. On the Start menu, click Run.
3. In the Open text box, type D:\setup.exe.

   where D: is the drive letter of your computer's CD-ROM drive.
4. Follow the on-screen prompts to complete the installation process.

Once installation of Pro-Motion is complete, insert the PMD Prodigy/CME SDK disk, and follow the same procedure above as for Pro-Motion installation.

Upon completion of the installation process for Pro-Motion and PMD Prodigy/CME SDK, the following components will be installed:

- Pro-Motion – an application for communicating to, and exercising the installed card. Refer to the *Pro-Motion User's Guide* for operating instructions.
- C-Motion – source code which may be used for developing motion applications in C/C++ based on the Magellan Motion Processor.
- Visual Basic Libraries – a DLL and example source code which may be used for developing motion applications in Visual Basic based on the Magellan Motion Processor.
- PDF versions of the included manuals. The Adobe Acrobat Reader is required for viewing these files. If the Adobe Acrobat Reader is not installed on your computer, it may be freely downloaded from http://www.adobe.com.

1.9 Preparing the Card for Installation

Figure 1-5 shows the location of the resistor packs JS1, JS2, JS3, along with other components such as Atlas sockets and connectors. These items are listed in the table below

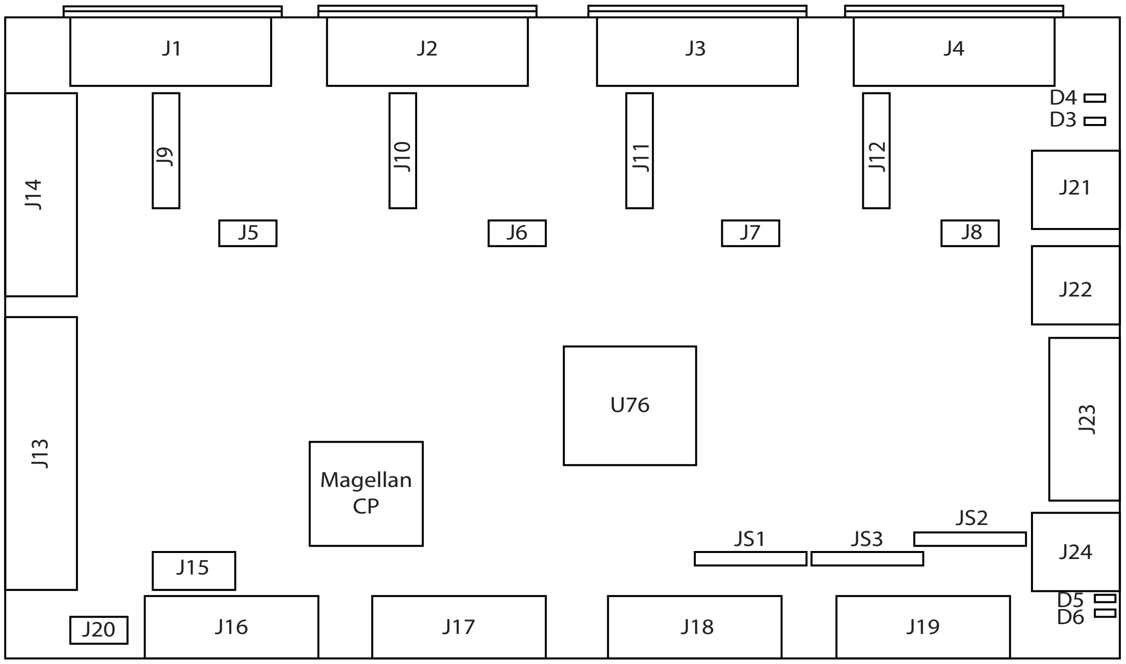


Figure 1-5: Prodigy/CME Machine-Controller Card Component Location

Card Components

The following table describes selected components on the card (as shown in Figure 1-5) and their functionality.

Label	Description
JS1-JS3	Resistor packs
J1, J2, J3, J4	Axis #1, 2, 3, and 4 Atlas sockets (respectively)
J5, J6, J7, J8	Axis #1, 2, 3, and 4 Motor Power connectors (respectively)
J9, J10, J11, J12	Axis #1, 2, 3, and 4 Motor Drive connectors (respectively)
J13	General I/O Connector
J14	Amplifier I/O Connector
J15	Expansion Connector
J16, J17, J18, J19	Axis #1, 2, 3, and Motor Signal connectors (respectively)
J20	+5V Power
J21, J22	CANbus1 and 2 Connectors (respectively)
J23	Serial Connector
J24	Ethernet Connector

1.9.1 Resistor Pack Settings

The Prodigy/CME Machine-Controller card has minimal user-adjustable settings. Most settings are software configurable. To prepare the card for installation, the user-specified resistor pack options should be checked, as described in the following table.

Item	Setting	Description
Resistor packs JS1, JS2, JS3	Installed; this is the default setting of resistor packs JS1–JS3.	If differential connections are being used, leave these resistor packs installed.
	Removed	If single-ended encoder connections are being used, remove the resistor packs.

1.9.2 L-Bracket Installation

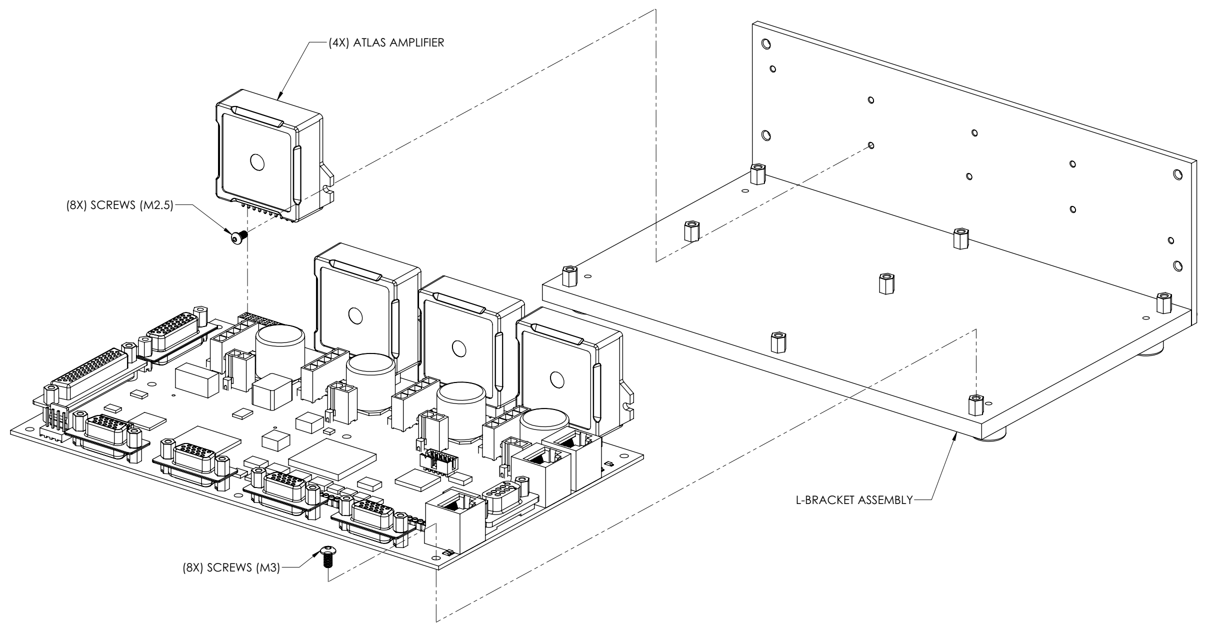
If you are not using an L-bracket, or if your machine controller card already has the Atlas amplifiers and L-bracket hardware installed you may skip to the next section.

Refer to Figure 1-6 to install either the base plate or combined base/vertical plate L-bracket hardware. All necessary fasteners should be included with the L-bracket hardware. You will need a 1.5 mm and a 2.0 mm allen key for installation of the L-bracket.

To install the base plate or the combined base/vertical plates use the screws as shown in Figure 1-6 to mount the machine controller card onto the base plate.

If attaching to the combined base/vertical L-bracket attach, but do not fully tighten, each fastener. It may be necessary to adjust the position of the card on the base plate when attaching Atlas units to the vertical plate. If attaching to the base plate only tighten the fasteners using moderate torque.

Figure 1-6: L-Bracket and Card Assembly



1.9.3 Atlas Amplifier Electrical Installation

If your machine controller card already has the Atlas amplifiers installed you may skip to Section 1.10, "Connection Summary."

Figure 1-7 shows the machine controller card along with the correct orientation of the Atlas amplifier while installing into the card. You will need to install all Atlas amplifiers that you plan to use with the machine controller card.

The motor type of the Atlas should conform to the motor type that will be utilized for that axis. For example if your system has a DC brush motor at axis #1, and a step motor connected at axis #2, you should install a DC brush motor Atlas in the axis #1 socket, and a step motor Atlas in the axis #2 socket.

If you are using the full L-bracket with vertical plate, begin by mounting the pre-cut thermal pads to each Atlas unit. Carefully align the pads with the Atlas' metallic backing, and apply finger pressure to adhere the pads to the metal. Once pressed in place the pads should stay in place, but if required the pads can be removed and remounted.

Next, to install the Atlas units into the socket, confirm that the Atlas is oriented correctly, with the metal heat sink surface facing toward the abutting edge of the card. Carefully align the Atlas pins to the socket and press firmly down until the Atlas is fully seated in the socket.

Extreme care should be taken when installing the Atlas into its socket. Failure to orient the Atlas correctly, or mis-alignment of pins may result in damage to the Prodigy/CME Machine-Controller card, Atlas units, or both.

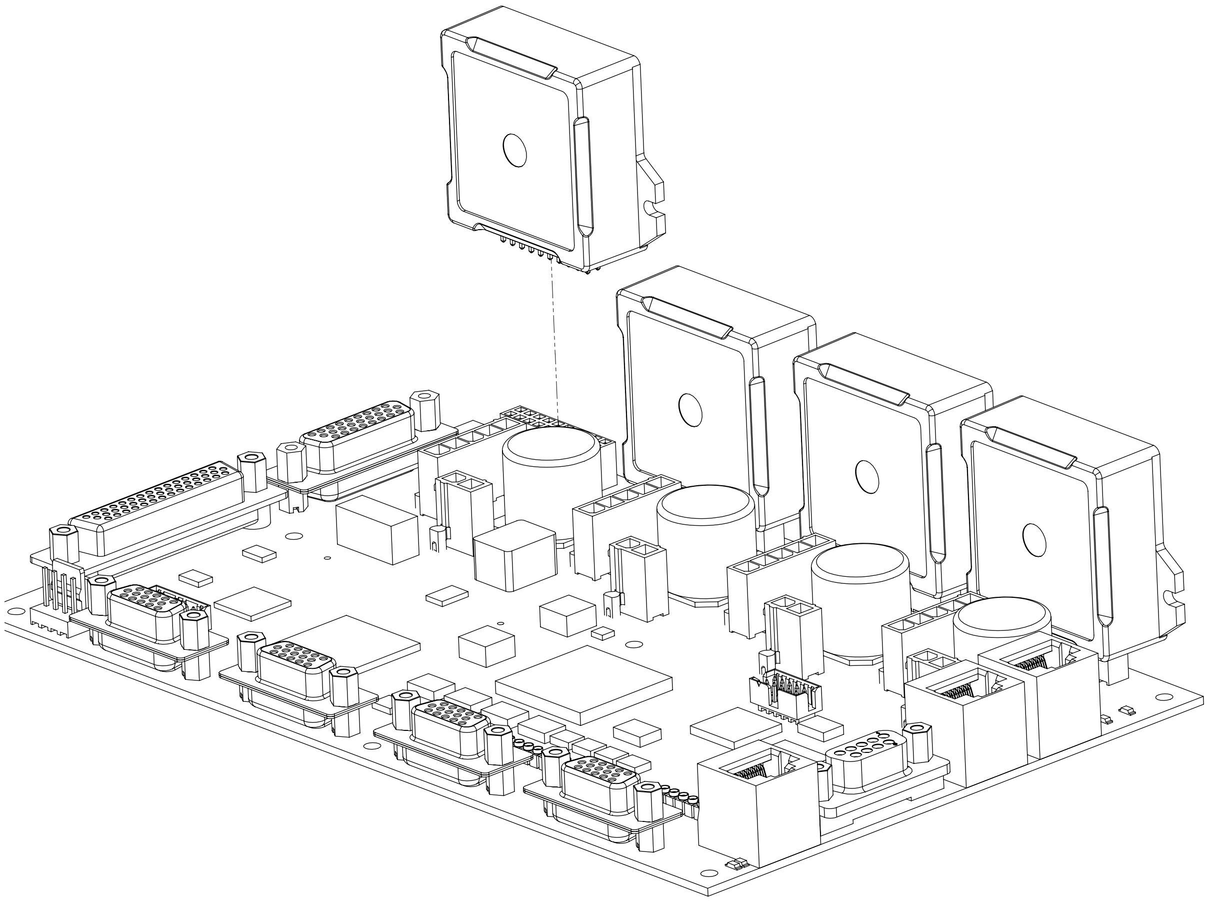


Figure 1-7: Atlas Installation into Machine-Controller Card

1.9.4 Atlas Amplifier Mechanical Installation

If your machine controller card already has the Atlas amplifiers installed you may skip to the next section.

With thermal pads installed on each Atlas, using Figure 1-6 as a reference, attach each Atlas unit to the vertical plate. Apply moderate torque to each fastener.

Once all Atlas units have been mechanically installed to the vertical plate, tighten the machine controller card base plate fasteners with moderate pressure.

Congratulations! You have now completed mechanical assembly of the L-brackets to the machine controller card and Atlas units.

1.10 Connection Summary

Figure 1-8, Figure 1-9, and Figure 1-10 provide general connection overviews for each machine controller axis. For each axis there is a feedback connector, a motor drive connector, and a motor power connector.

Figure 1-8: With Brushless DC Motor

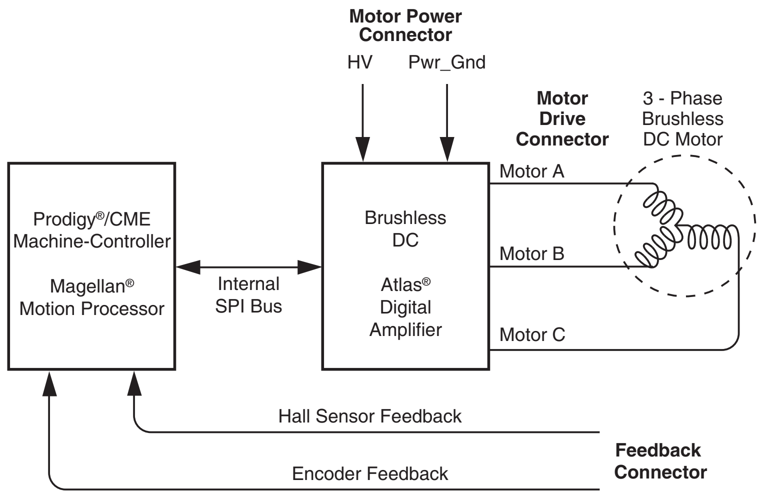
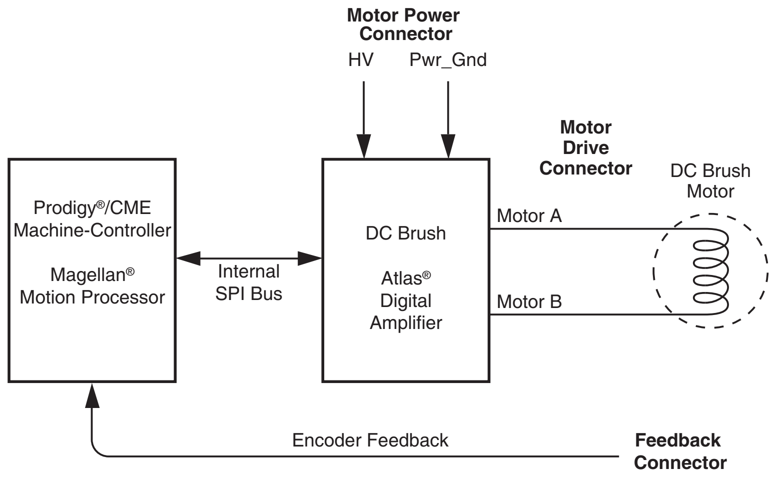


Figure 1-9: With DC Brush Motor



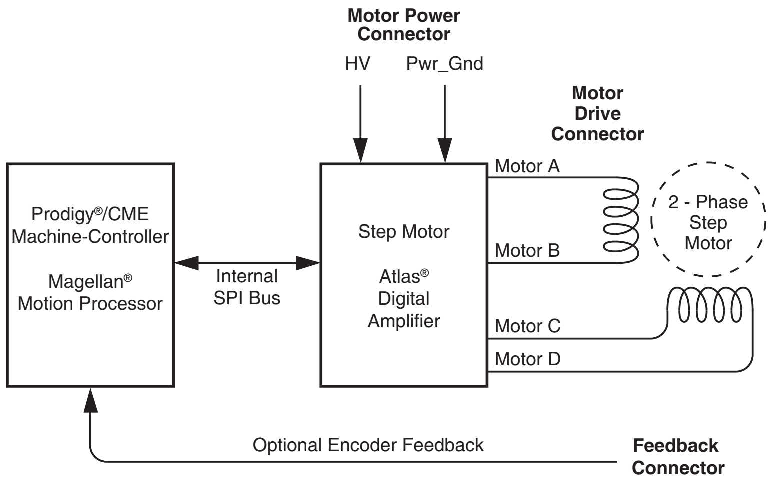


Figure 1-10: With 2-Phase Step Motor

You will need signals on each of these connectors to properly install the machine controller card with the attached motors. The following three sections detail how this should be done.

1.10.1 Motor Feedback Connections

The following table summarizes the motor signal connections to the Prodigy/CME Machine-Controller. Between one and four axes may be connected, depending on the specific machine controller card and application requirements. All connections are made through the Motor Signal Connectors, which are high density female DB-15s.

If you have ordered the developer's kit version of the machine controller, you may find it convenient to use the provided DB 15 stub cable sets to connect to the motor signals. There is one connector for each axis, and the signals are identical for all axes:

Pin #	Signal Name	Description
1	QuadA+	Differential A+ quadrature input. (*optional for step motor axes)*
2	QuadA-	Differential A- quadrature input. (*optional for step motor axes)*
3	QuadB+	Differential B+ quadrature input. (*optional for step motor axes)*
4	QuadB-	Differential B- quadrature input. (*optional for step motor axes)*
5	GND	This is the preferred ground connection for the quadrature and Index signal inputs
6	Index+	Differential Index+ quadrature input. (*optional for step motor axes)*
7	Index-	Differential Index- quadrature input. (*optional for step motor axes)*
8	HallA	Hall signal input phase A. *not used for DC brush or step motors*
9	HallB	Hall signal input phase B. *not used for DC brush or step motors*
10	HallC	Hall signal input phase C. *not used for DC brush or step motors*
11	Home	Home signal input (*optional*)
12	PosLim	Positive position limit input (*optional*)
13	NegLim	Negative position limit input (*optional*)
14	+5V	+5V power output which may be used to power the motor's encoder circuitry
15	GND	This is the preferred ground connection for the +5V output along with the Hall, home, and limit input signals.

If using single-ended encoder connection refer to Section 2.2.3.2, "Connections & Associated Signals," for wiring information.

J16, J17, J18, and J19 are the motor signal connectors for axes 1, 2, 3, and 4 respectively. They may be located on the card using Figure 1-5.

1.10.2 Motor Drive Connections

The following table summarizes the motor drive connections from the Prodigy/CME Machine-Controller to the coils of your motor. Each motor drive connector is designed to connect to all available motor types; brushless DC, DC brush, step motor. There are four motor drive connections for each axis and a shield connection. Not every motor type uses all four drive connections however.

Between one and four axes may be connected depending on the specific card purchased and application requirements. All connections are made through the Motor Drive Connectors, which are male Molex Mini-Fit Plus style connectors.

If you have ordered the developer's kit version of the machine controller, you may find it convenient to use the provided motor drive stub cable sets to connect to the motor drive signals. There is one connector for each axis, and the signals are identical for all axes:

Pin #	Signal Name	Description
1	Motor A	A motor drive lead. Used with all motor types.
2	Motor B	B motor drive lead. Used with all motor types
3	Motor C	C motor drive lead. Used with all motor types except DC brush
4	Motor D	D motor drive lead. Used with step motors only
5	Shield	Connection to motor ground. A shield connection is strongly recommended, however not required for most motor setups.

You may refer to Figures 1-8, 1-9, and 1-10 or use the table below to determine which leads should be connected for each supported motor type:

Motor type	Machine Controller Motor Lead	Motor Coil Connections
Brushless DC	Motor A	A winding connection
	Motor B	B winding connection
	Motor C	C winding connection
	Shield	(optional) motor shield connection
DC Brush	Motor A	+ winding connection
	Motor B	- winding connection
	Shield	(optional) motor shield connection
Step motor	Motor A	phase A+ winding connection
	Motor B	phase A- winding connection
	Motor C	phase B+ winding connection
	Motor D	phase B- winding connection
	Shield	(optional) shield connection

J9, J10, J11, and J12 are the motor signal connectors for axes 1, 2, 3, and 4 respectively. They may be located on the card using Figure 1-5.

Shield connections to the motor are not required but are highly recommended. Not connecting the shield signal for each axis may result in increased EMI (electromagnetic interference), and reduced immunity to ESD (electro static discharge).

1.10.3 Motor Power Connections

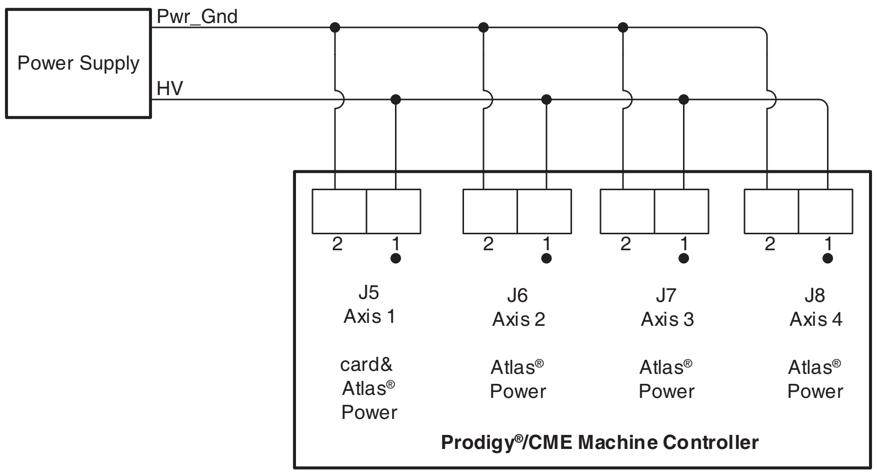


Figure 1-11: Power Connections

The following table summarizes the motor power connections from the Prodigy/CME Machine-Controller to your power supply. Each motor driven by an Atlas amplifier must have separate power provided to it. Although most applications will power each axis at the same voltage from a common supply, different voltages may be connected if desired.

In addition, axis 1 motor power should always be provided whether or not an Atlas is installed at that axis. Axis 1 is the power connection from which the card logic power is derived using on-board DC-DC converters. Note that under some circumstances it may be desirable to provide just the +5V card logic power. This can be done via connector J20, however power should only be applied at J20 if no power is applied to Atlas #1.

Between one and four axes may be powered depending on the specific machine controller card and application requirements. All connections are made through the Motor Power Connectors, which are male two- conductor Molex Mini-Fit Plus 2-style connectors.

If you have ordered the developer's kit version of the machine controller, you may find it convenient to use the provided motor power stub cable sets. There is one connector for each axis, and the signals are identical for all axes:

Pin #	Signal Name	Description
1	HV	Positive motor voltage power
2	Pwr_Gnd	Motor voltage power ground

J5, J6, J7, and J8 are the motor power connectors for axes 1, 2, 3, and 4 respectively. They may be located on the card using Figure 1-5.

See Chapter 3, "Electrical Reference" for complete machine controller electrical specifications.

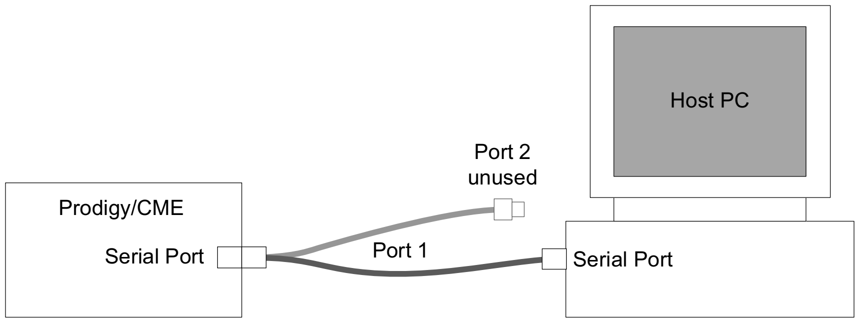
1.10.4 Communication Connections

While the Prodigy/CME Machine-Controller card can communicate using Ethernet, CANbus, and one of two serial modes (RS-232 and RS-485), in this first-time installation we will set up the card for Ethernet communications. To set up the card for operation in other communication modes, see Chapter 2, "Operation" and the *Pro-Motion User's Guide.*

Because the Ethernet address for the card will need to be configured, we will first connect the RS-232 serial port, and then the Ethernet port on the card. During initialization, we will then use Pro-Motion to change the IP address that the card expects Ethernet communications on.

If the standard PMD accessory cables are used, the base of the "Y" of the dual serial cable (PMD p/n Cable-4355-01.R) should be connected to the Prodigy/CME Machine-Controller card's J23 Serial Connector, while the opposite end of the serial cable marked "Port1" should be connected to your computer's 9 pin serial port. If your computer does not have a dedicated serial port, a standard USB to serial converter should be used. Figure 1-12 shows a typical serial port connection.

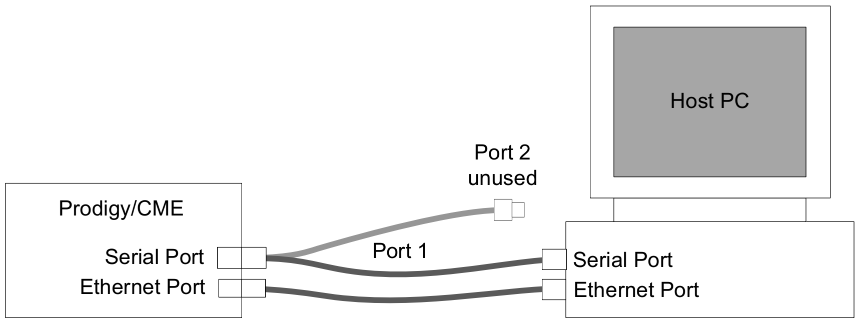
Figure 1-12: Serial Port Connection



The Ethernet connection will not be made until serial communications are established. See Section 1.12.2, "Setting the Ethernet Parameters," for detailed instructions on when to physically make the Ethernet connection. When Ethernet is ready to connect, use PMD's Cable-RJ45-02-R, and plug one end of the connector into J24, and the other end into your computer's Ethernet port. Figure 1-13 shows the complete serial and Ethernet connection setup.

The communication connections are locatable on the card using Figure 1-1.

Figure 1-13: Serial and Ethernet Connection



1.11 Applying Power

Once you have made your motion hardware, communication, and power connections, hardware installation is complete and the card is ready for operation. When power is applied, the Prodigy/CME Machine-Controller's green power LED should light. This LED is locatable as D4 using Figure 1-5. If the LED does not light, recheck the connections.

After power up, no motor output will be applied. Therefore, the motors should remain stationary. If the motors move or jump, power down the card and check the motor and encoder connections. If anomalous behavior is still observed, call PMD for assistance.

1.12 First-Time System Verification

The first time system verification procedure summarized below has two overall goals. The first is to connect the Prodigy card with the PC that is being used so that they are communicating properly, and the second is to initialize each axis of the system and bring it under stable control capable of making trajectory moves. While there are many additional capabilities that Pro-Motion and the Prodigy motion cards provide, these steps will create a foundation for further, successful exploration and development.

Here is a summary of the steps that will be used during first time system verification. Each of these steps will be described below in a separate manual section.

1. Initiate Pro-Motion and establish communication between the PC and the card using the serial communications link.
2. Use the serial communications link to change the Ethernet communications parameters of the Prodigy/CME Machine-Controller card.
3. Establish communication between the PC and the card using the Ethernet communications link.
4. Disconnect serial communications.
5. Run Pro-Motion's Axis wizard for each axis of your system to initialize parameters such as encoder direction and safe servo parameters (if using a servo motor).
6. Execute a simple trajectory profile on each axis demonstrating that it is operating correctly and under stable control.

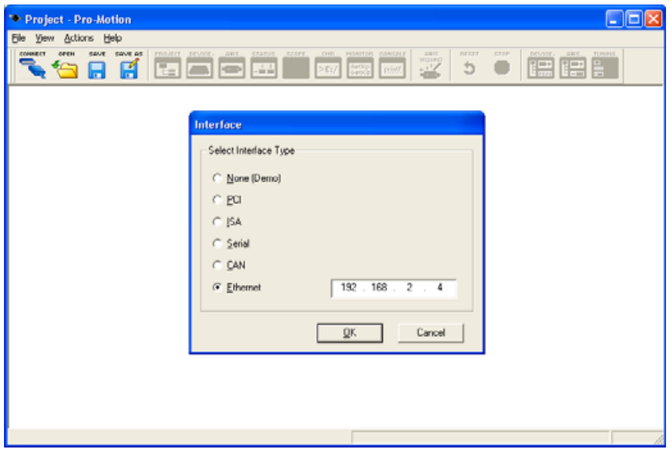
During this first time system setup you may find it useful to refer to other PMD manuals including the *Pro-Motion User's Guide* for complete information on the Pro-Motion application. You may also want to refer to the *Magellan Motion Processor User's Guide* to familiarize yourself with operation of the Magellan Motion Processor, which lies at the heart of all Prodigy Motion cards.

1.12.1 Establishing Serial Communications

To establish serial communications:

1. Make sure the Prodigy card is powered and connected to the PC via its serial port.
2. On the Start menu, click the Pro-Motion application.

   When Pro-Motion is launched you will be prompted with an Interface selection window. A typical screen view when first launching Pro-Motion appears below.



Select Interface Type
( ) None (Demo)
( ) PCI
( ) ISA
( ) Serial
( ) CAN
(•) Ethernet 192 . 168 . 2 . 4

3. Click the Connect icon on the toolbar.

Alternatively, on the File menu, click Connect.

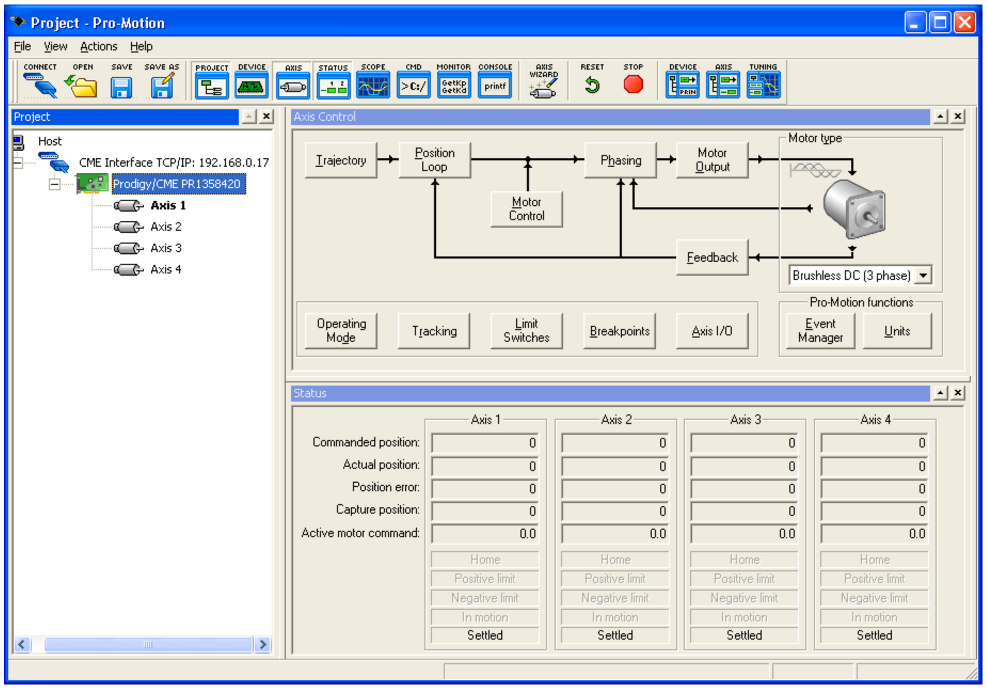
The purpose of the Interface dialog box is to indicate to Pro-Motion how your Prodigy/CME Machine-Controller card is connected to the PC. It provides various selectable communication options such as PCI, serial, CANbus, Ethernet.

4 Click Serial, and then click OK.

The Serial Port dialog box displays with default communication values of 57,600 baud, no parity, 1 stop bit, and point to point protocol.

5 Click OK without changing any of these settings.

If serial communication is correctly established, a set of object graphics loads into the Project window to the left, as shown in the following figure.



Host
CME Interface TCP/IP: 192.168.0.17
Prodigy/CME PR1358420
Axis 1
Axis 2
Axis 3
Axis 4

Motor type: Brushless DC (3 phase)

	Axis 1	Axis 2	Axis 3	Axis 4
Commanded position:	0	0	0	0
Actual position:	0	0	0	0
Position error:	0	0	0	0
Capture position:	0	0	0	0
Active motor command:	0.0	0.0	0.0	0.0
	Home	Home	Home	Home
	Positive limit	Positive limit	Positive limit	Positive limit
	Negative limit	Negative limit	Negative limit	Negative limit
	In motion	In motion	In motion	In motion
	Settled	Settled	Settled	Settled

For example for a four axis Prodigy card, you see the card name next to an icon of a card, and below that you see four axis icons, one for each available axis of the motion card. Highlighting (single clicking) either the card icon or one of the axis icons with the mouse is used to select specific cards or axes, and is useful later on in the first time system verification.

If serial communications are not correctly established, after approximately 2 seconds a dialog box appears indicating that a Communications Timeout Error has occurred. If this is the case, recheck your connections and repeat from step 1 above. If after repeated attempts a connection can still not be established, call PMD for assistance.

1.12.2 Setting the Ethernet Parameters

To set the Ethernet parameters:

1 With serial communications functioning properly, click the Device toolbar button.

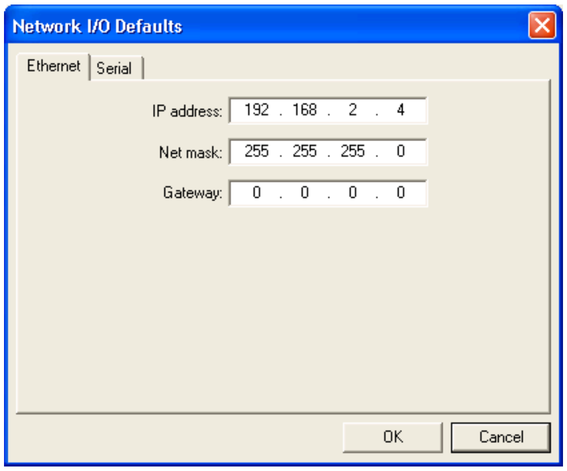
The Device window appears.

2 Click Network I/O.

The Network I/O Defaults dialog box appears.

3 Click the Ethernet tab.

The Ethernet tab appears with data entry fields for the IP Address, the Net Mask, and the Gateway.



IP address: 192 . 168 . 2 . 4
Net mask: 255 . 255 . 255 . 0
Gateway: 0 . 0 . 0 . 0

For a typical installation, you will not change the Net mask and Gateway default values, but you must specify a valid, and unique, IP Address for the card to be located on your Ethernet network. If you are not sure what IP addresses are free and available for your Ethernet network, contact your system administrator.

4 Enter the IP Address in the corresponding data field as well as the net mask and gateway, if this is required for your network.

5 Click OK to store as the power on default.

6 Click the Reset toolbar button.

After the card is reset, it uses the default parameters that you specified.

7 Connect the Ethernet cable.

See Section 1.10.4, "Communication Connections" for details.

The card is now ready for Ethernet communications.

1.12.3 Establishing Ethernet Communications

The card's IP Address has now been set, but Pro-Motion does not as yet know what IP address it should use for Ethernet communications to the card. To establish Ethernet communications:

1 Click the Connect toolbar button.

2 Select Ethernet, and then click OK.

3 Enter the same IP Address as was specified for the Prodigy/CME Machine-Controller card.

4 When complete, click OK.

If Ethernet communications are successful, an additional set of graphical icons representing your card and axes will be loaded into the Project window to the left below the first set created while establishing communications by serial link.

If communication is not successful, after about 30 seconds, a Communications Timeout Error dialog box appears. If this is the case recheck your connections, and retry to establish Ethernet communications. See step 1 in Section 1.12.2, "Setting the Ethernet Parameters" for details.

With Ethernet communications functioning properly, the final step is to disable serial communications.

1.12.4 Disconnecting Serial Communications

To disconnect serial communications:

1 Select the serial link version of the Prodigy/CME Machine-Controller card in the Project window to the left.

2 Click the Disconnect toolbar button.

A dialog box appears asking if you are sure you want to disconnect.

3 Click OK.

You will notice that the serial Prodigy/CME Machine-Controller card icon and axes graphical icons in the Project box disappear, leaving only the Ethernet link icons for the card and axes.

Congratulations, Ethernet communication is now up and running, and you are ready execute any of the functions on Pro-Motion via Ethernet.

Multiple Pro-Motion users can connect to the same Prodigy/CME Machine-Controller card on TCP port 40099. Up to four simultaneous connections can be made. There are various situations where this may be useful. For example one PC can function as a 'monitoring station' for a particular Prodigy/CME card while another PC provides commands to that same card. Be aware however that two or more users sending motion commands to the same card can cause unexpected motion, and should be avoided.

When connecting your Prodigy card for use on an Ethernet network, be sure that the IP address provided for the Prodigy card does not conflict with the addresses of other users on the network. See Section 1.12.2, “Setting the Ethernet Parameters” for a description of changing the IP address.

1.12.5 Initializing Motion Axes

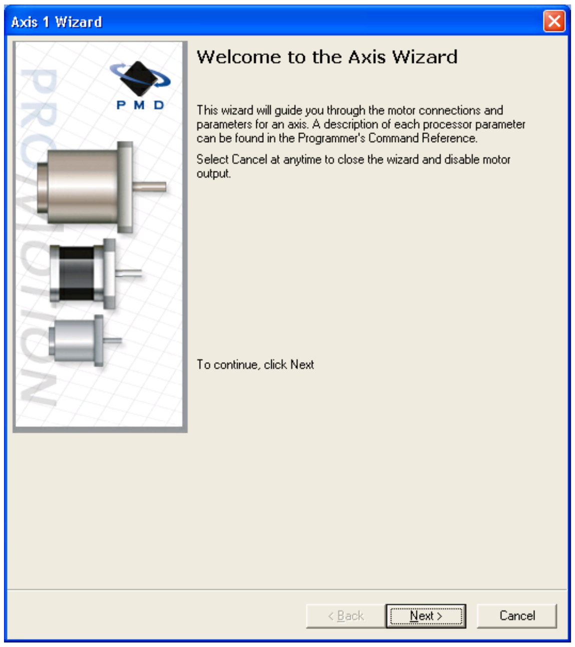
The next step to verify the correct operation of the system is to initialize each axis of the motion system sequentially, thereby verifying correct amplifier connection, encoder feedback connections (if an encoder is used), and other motion functions. All of this can be conveniently accomplished using Pro-Motion’s Axis Wizard function. This versatile and easy to use tool initializes all supported motor types including step, DC brush, and brushless DC.

To operate the Axis Wizard:

1 Click to select the axis icon that you would like to initialize (normally this would be Axis #1) in the Project window to the left of the screen.

2 With this icon highlighted, click the Axis Wizard toolbar button.

The Axis Wizard initialization window appears.



Welcome to the Axis Wizard

This wizard will guide you through the motor connections and parameters for an axis. A description of each processor parameter can be found in the Programmer's Command Reference.

Select Cancel at anytime to close the wizard and disable motor output.

To continue, click Next

3 Click Next and follow the Axis Wizard instructions for each page of the axis initialization process.

A typical Axis Wizard sequence takes 3-5 minutes. If you have specific questions about the Axis Wizard, refer to the *Pro-Motion User's Guide* for detailed information.

Upon a normal completion of the Axis Wizard, the axis will be ready to make a controlled move.

The most common reasons for the Axis Wizard to not complete normally are an inability to auto-tune the servo motor, or problems determining the correct commutation sequence for brushless DC motors when commutated by the Magellan Motion Processor. Should this happen, it is possible to perform a manual tuning or commutation setup if desired. Refer to the *Pro-Motion User's Guide* for more information, or call PMD for technical assistance.

The Axis Wizard auto tuning routine, which is used with servo motors, is designed to provide stable, but not optimal, parameters for motion. Pro-Motion provides a wealth of functions including a high speed hardware trace oscilloscope that can assist you in determining optimal servo parameters. Values provided by the Axis Wizard during auto tuning may or may not be safe for your system, and it is up to the user to determine if and when they should be used.

1.12.6 Performing a Simple Trajectory Move

The last step in first time system verification is to perform a simple move for each axis. To perform a simple move:

1 In the Project Window, select the motion axis that you would like to move by clicking the corresponding icon.

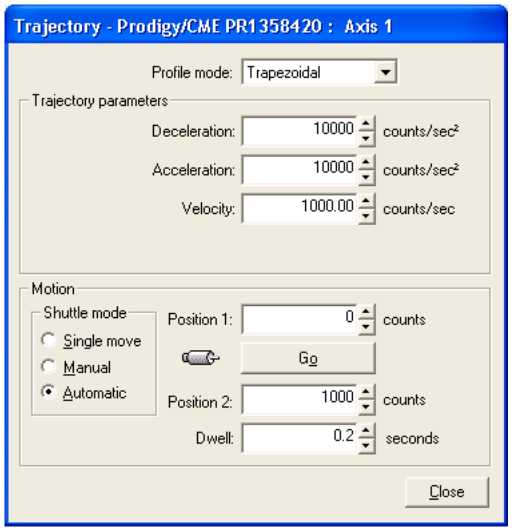
2 Click the Axis view button on the far right of the toolbar.

Alternatively, click Axis View on the Axis menu.

Your screen organization changes to give easy access to windows that are used while exercising the motion axes.

3 Click the Trajectory button in the Axis Control window.

The Trajectory dialog box appears.



Trajectory - Prodigy/CME PR1358420 : Axis 1

Profile mode: Trapezoidal

Trajectory parameters

Deceleration: 10000 counts/sec²

Acceleration: 10000 counts/sec²

Velocity: 1000.00 counts/sec

Motion

Shuttle mode: Single move, Manual, [✓ Automatic]

Position 1: 0 counts

Position 2: 1000 counts

Dwell: 0.2 seconds

4 In the Profile mode list, select Trapezoidal.

5 Enter motion profiles for deceleration, acceleration, velocity, and destination position (Position 1) that are safe for your system and will demonstrate proper motion.

Pro-Motion provides various selectable units for distance and time, but defaults to units of encoder counts (or pulses for step motors) for distance and seconds for time. This means the default units for velocity are counts/sec, and the default units for acceleration and deceleration are counts/sec^2. So for a motor that has 2,000 counts per rotation, to perform a symmetric trapezoidal move of 25 rotations with a top speed of 5 rotations per second and with an acceleration time of two seconds, the parameters in the Trajectory dialog box would be set as follows:

Deceleration: 5,000 counts/sec^2

Acceleration: 5,000 counts/sec^2

Velocity: 10,000 counts/sec

Position 1: 0 counts

Position 2: 50,000 counts

6 Click Go and confirm that the motion occurred in a stable and controlled fashion.

Congratulations! First time system verification for this axis is now complete. You should now initialize all of the axes in your system. Go to Section 1.12.5, “Initializing Motion Axes” and repeat the steps.

1.13 Developing User Application Code

Pro-Motion provides an intuitive, convenient, graphical interface to setup and exercise motion controllers such as Prodigy/CME cards or ION® Digital Drives. Eventually though, you will begin to design application-specific software code that will serve as your system controller. PMD supports two standard languages to accomplish this; Visual Basic, through the provided DLLs, and C/C++, through the source code-based system known as C-Motion.

For more information on Visual Basic support and C-Motion, please consult the *Magellan Motion Processor Programmer's Command Reference* and the *PMD Resource Access Protocol Programmer's Reference.*

1.13.1 Architecture

Figure 1-14 shows two ways to locate your application code with the Prodigy/CME Machine-Controller cards.

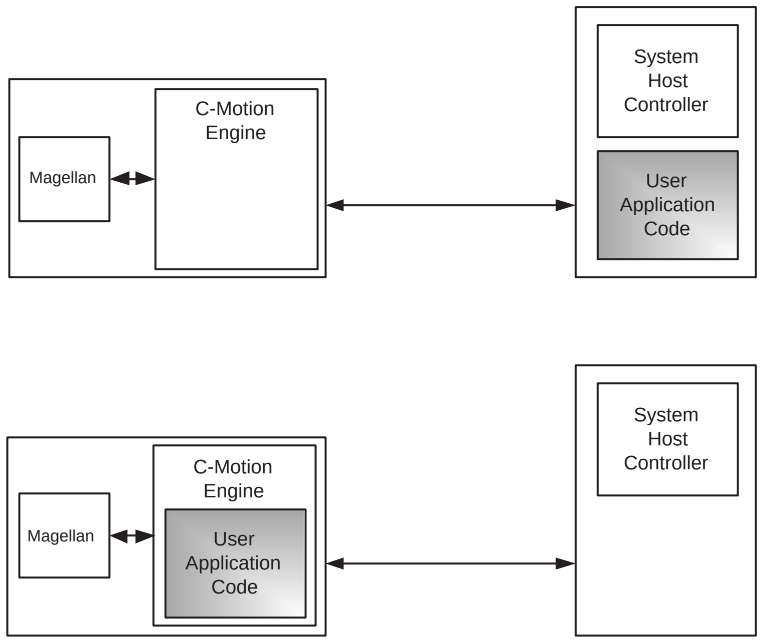


Figure 1-14: Two Ways to Locate the Code on the Prodigy/CME Machine-Controller Card

When located on a host controller, the user's code communicates via the serial, CANbus, or Ethernet link to the Prodigy/CME Machine-Controller card.

Either Visual Basic or C-Motion can be used to communicate to the Prodigy/CME card, and the choice of software tools to compile and debug C code is typically determined by the developer. The advantages of a 'host-centered' machine controller approach are that software sequences can be centralized, and the user's code has convenient access to the PC's keyboard, mouse, or touch screen user interface facilities.

When located in the Prodigy/CME Machine-Controller card, the user's code communicates directly to the resources available on the card such as the Magellan Motion Processor. This has speed advantages both in communicating with those resources, and in real time code execution predictability.

Another feature of locating code on the card is that the C-Motion Engine can be programmed to receive or send commands from a higher level controller via the Prodigy/CME's serial, CANbus, or Ethernet links. In this way the user's application code, downloaded onto the card, forms a local controller while still communicating to the higher level controller that synchronizes overall system behavior.

Finally, locating the code in the card allows use of the Prodigy/CME card as a fully standalone controller. In this mode a host controller network communication link is not used, and one or more of the card's network or digital I/O ports are typically used to interface to buttons or other devices. This is also a common configuration for machines that will interface to a central PLC (Programmable Logic Controller).

By supporting application code on the host controller as well as downloaded directly on the card, the user is provided with multiple options for optimizing the control architecture of their machine, and locating their software on the hardware platform that will best match their machine's operational and performance requirements.

1.13.2 C-Motion Engine

The *C-Motion Engine Development Tools* manual provides a complete description of how to create C-Motion code that can be downloaded onto the Prodigy/CME Machine-Controller's C-Motion Engine.

The C-Motion Engine development environment operates on the PC. Code is edited, compiled, linked, downloaded, and monitored via programs that reside on the PC. Systems which have high level PC-based code concurrently sending commands to the Prodigy card can locate that code on the same PC as the one used for C-Motion code development, or on a separate PC.

All of these considerations and much more are discussed in the *C-Motion Engine Development Tools Manual*, which includes a convenient Getting Started section that introduces the C-Motion Engine IDE (Integrated Development Environment) and walks you through an example session resulting in code being downloaded and executed on the Prodigy/CME Machine-Controller card.

2. Operation

In This Chapter

The Prodigy/CME Machine-Controllers are high-performance card based motion controllers for DC brush, brushless DC, and step motors. They are based on Magellan Motion Processors, which perform motion command interpretation and numerous other real-time functions. They directly host Atlas digital amplifiers for each axis, eliminating the need for off-card amplifiers.

Machine controller cards include a powerful C-Motion Engine module (CME) which allows C-language application code to be downloaded and executed directly on the card. To assemble a complete functioning system the only additional components needed are motors, cables, and a power supply.

The following diagram provides a functional block diagram of the Prodigy/CME Machine-Controller card:

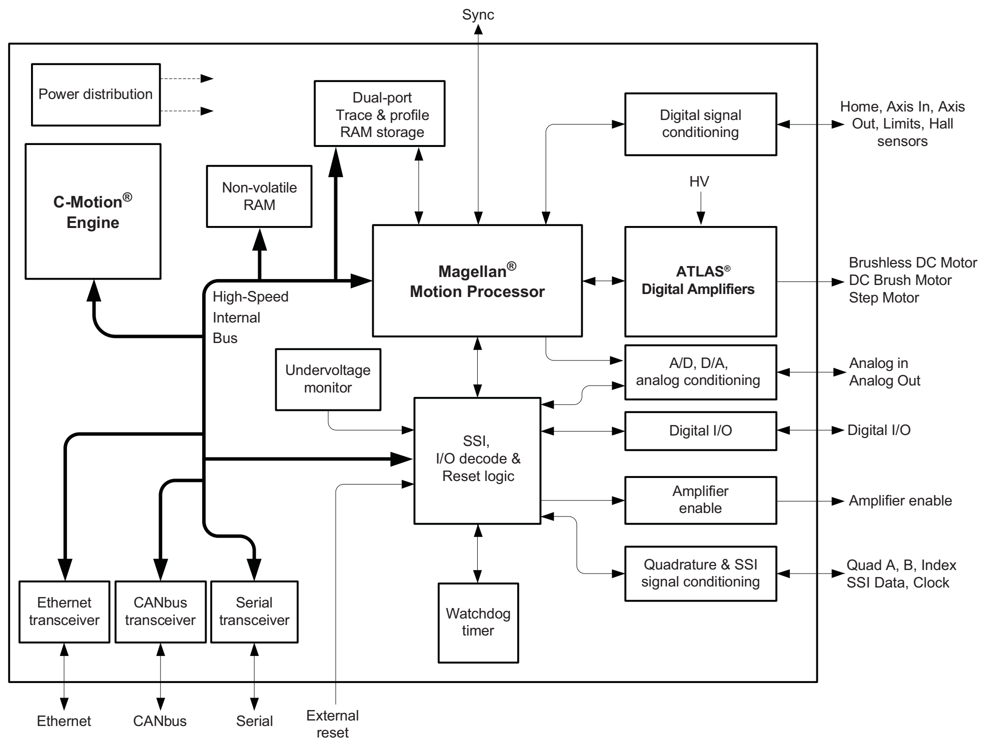


Figure 2-1: Prodigy/CME Machine-Controller Card Internal Block Diagram

2.1 Card Function Summary

Prodigy/CME Machine-Controller card functions can be broken down into six overall categories:

Magellan functions - These are functions which reside in the Magellan Motion Processor. Included are profile generation, DC brush and brushless DC servo loop closure, breakpoint processing, and much more. These functions are accessed through the Magellan Motion Processor's command set, which allows for sophisticated control of the motion axes and associated hardware.

I/O functions - These are digital and analog input and output functions. There are 16 general purpose digital channels, consisting of four dedicated inputs, four dedicated outputs, and eight bi-directional. There are eight general purpose +/- 10V analog input channels and eight general purpose +/- 10V output channels.

Communications functions - The Prodigy/CME Machine-Controller card provides sophisticated communication facilities consisting of two serial ports, a CANbus port, and an Ethernet port.

Atlas Amplifier functions - The on-card Atlas amplifiers accept motor output commands from the Magellan motion processor and provide high performance current control and amplification for attached step, DC brush, or brushless DC motors. There are numerous programmable parameters including current gains, safety parameters, and more.

General Card functions - These are general purpose card functions such as a dual-ported RAM, a non-volatile RAM, a card reset function, a card watch-dog timer, and several others.

C-Motion Engine functions - The C-Motion Engine is a self-contained, high performance code execution unit that allows C-Motion code to be downloaded and executed on the Prodigy/CME Machine-Controller card. It can communicate with various resources on the card including the Magellan Motion Processor, the card's serial, CANbus, and Ethernet ports, and other on-card resources such as the dual-ported RAM.

2.1.1 Card Access Basics

Access to the Prodigy/CME Machine-Controller card from the serial, CANbus, or Ethernet ports is provided by a protocol called the *PMD Resource access Protocol* (PRP). This easy-to-use yet powerful system utilizes actions, resources, and addresses to access the Prodigy/CME Machine-Controller card's functions. Various card functions are organized into resources, and resources process actions sent to them. Actions can send information, request information, or command specific events to occur. Addresses allow access to a specific resource on the card, or connected to the card, via the serial, CANbus, or Ethernet connections.

A basic communication to the Prodigy/CME Machine-Controller card consists of a 16-bit PRP header, and an optional message body. The message body contains data associated with the specified PRP action, but some actions do not require a message body. After a PRP communication is sent to the card, a return communication is sent by the Prodigy/CME Machine-Controller card which consists of a PRP header and an optional return message body. The return message body may contain information associated with the requested PRP action, or it may contain error information if there was a problem processing the requested action.

There are five different resource types supported by the Prodigy/CME Machine-Controller card. The ***Device*** resource indicates functionality that is addressed to the entire card, the ***MotionProcessor*** resource indicates a Magellan Motion Processor, the ***CMotionEngine*** resource indicates the C-Motion Engine, the ***Memory*** resource indicates the dual-ported RAM and the non-volatile RAM (Random Access Memory), and the ***Peripheral*** resource indicates a communications connection, the digital and analog I/O functions, or other on-card registers.

There are ten different PRP actions including Command, which is used to send commands to resources such as the Magellan Motion Processor, Send and Receive, which are used to communicate using the serial, CANbus, and Ethernet ports, Read and Write, which are used to access memory-type devices such as the on-card dual-ported RAM and the non-volatile RAM, and Set and Get, which are used to load or read parameters.

For complete information on the format and function of PMD Resource Access Protocol refer to the *PMD Resource Access Protocol Programmer's Reference.*

Although it may be useful to be familiar with the machine controller's command processing architecture, most users will not need to know all of these details. Most users will prototype motion sequences using PMD's Pro-Motion Windows-based motion development application, and will use PMD's high-level C-Motion library to develop user application code. C-Motion provides many direct function calls which hide the low level details of communication and card architecture so that the user may directly access the card functions of interest to them.

In the subsequent sections of this chapter the C-Motion library functions used to access the available card functions will be provided along with the descriptions of the card functions themselves.

2.1.2 Peripheral I/O Space

Some of the machine controller's bit and word-oriented resources are accessed through what is known as the Peripheral I/O address space, also called PIO for short. This includes analog input and output, digital input and output, and various bit-oriented control registers.

The following table shows the machine controller's Peripheral I/O address map:

Address	Function	Read/Write	Comments
0x100, 0x120, 0x140, 0x160	SSI configuration register	Read & Write	0x100 is a bit-encoded register that controls the axis 1 SSI format, SSI enable/disable status, and SSI clock direction. 0x120 controls for axis 2, etc… up to 0x160 which controls these parameters for axis 4.
0x102, 0x122, 0x142, 0x162	SSI resolution register	Read & Write	0x102 is a 16-bit register that holds the SSI Absolute encoder word resolution for Axis 1. 0x122 holds the resolution for axis 2, etc… up to 0x162 which holds the resolution for axis 4.
0x104, 0x124, 0x144, 0x164	SSI frequency register	Read & Write	0x104 is a 16-bit register that holds the SSI clock frequency for Axis 1. 0x122 holds the frequency for axis 2, etc… up to 0x162 which holds the frequency for axis 4.
0x200	General purpose digital input values	Read	The 12 available input bits are stored in the low 12 bits of this register.
0x210	General purpose digital output write value	Read & Write	The 12 available output bits are stored in the low 12 bits of this register.
0x212	General purpose digital output write mask	Read & Write	The 12-bit output mask is stored in the low 12 bits of this register.
0x218	Amplifier enable signal write value	Read & Write	The 4 available output bits are stored in the low 4 bits of this register.
0x21A	Amplifier enable signal write mask	Read & Write	The 4 available output bits are stored in the low 4 bits of this register.
0x220	General purpose digital output signal command	Read	The commanded output states of the 12 output signals can be read in the low 12 bits of this register.

Address	Function	Read/Write	Comments
0x300 - 0x30E	General purpose analog output channel values for channels 1 through 8.	Read & Write	0x300 holds the channel 1 16-bit word, 0x302 holds the channel 2 16-bit word, etc… up to 0x30E which holds the channel 8 16-bit output word
0x310 - 0x31E	Analog output source selection	Read & Write	0x310 holds the analog output source (either Magellan or PRP) for channel 1, 0x312 for channel 2, etc… up to 0x31E which holds the channel 8 output source.
0x320	Analog output enable	Read & write	When set to 1, the low bit of this register enables analog channel output. When set to 0, the analog output voltages are shunted to 0.0 volts.
0x340 - 0x34E	Analog input channel values for channels 1 through 8.	Read	0x340 holds the channel 1 16-bit input word, 0x342 holds the channel 2 16-bit input word, etc… up to 0x34E which holds the channel 8 input word.

Various standard C-Motion Peripheral access commands such as PeriphOpen, PeriphRead, and PeriphWrite are used to access the contents of the registers located in the Peripheral I/O address space. The specific C-motion commands required are discussed along with each of the card functions introduced in the sections below.

2.2 Magellan Functions

The Magellan Motion Processor in Figure 2-1 forms the core of the Prodigy/CME Machine-Controller cards. Here is an overview of the functions provided by the Magellan Motion Processor:

- Profile generation
- Quadrature encoder processing and index capture
- DC brush and brushless DC servo loop closure
- Breakpoint processing
- AxisIn and AxisOut signal processing
- Trace
- Motion error detection, tracking windows, and axis-settled indicator
- Limit switch processing
- Atlas amplifier interfacing
- Analog amplifier output signal generation

The Magellan Motion Processor interfaces with motion hardware components such as feedback encoders and the on-card Atlas amplifiers directly through its own pin connections or through various signal conditioning circuitry. The following sections will provide information on all of these functions.

Magellan instructions are encoded in packets, which are sent to and from the Magellan Motion Processor. The Magellan processes these packets, performs requested functions, and returns requested data. Generally speaking each command packet has its own C-Motion command associated with it.

Within the Prodigy/CME Machine-Controller card the Magellan uses its high-speed parallel-word communications mode to connect to the card's internal communications bus, which allows the Magellan to be controlled via the C-Motion Engine, or via an external host controller connected to the Prodigy/CME Machine-Controller card by serial, CANbus, or Ethernet port.

2.2.1 Example Magellan Instructions

The Magellan instruction set is very flexible and powerful. The following example, which sets up and executes a simple trapezoidal profile, illustrates just a small part of the overall command set. In addition, this small sequence will introduce you to the look and feel of C-Motion, PMD's C-language callable motion interface that handles communications to and from all machine controller card resources.

In the Magellan user's guide commands are called instructions, and are often denoted via mnemonics. The sequence that we will execute is shown below in mnemonic notation, with comments included after the '//'

```
SetProfileMode Axis1, trapezoidal        // set profile mode to trapezoidal for axis 1
SetPosition Axis1, 12345                 // load a destination position for axis 1
SetVelocity Axis1, 223344                // load a velocity for axis 1
SetAcceleration Axis1, 1000              // load an acceleration for axis 1
SetDeceleration Axis1, 2000              // load a deceleration for axis 1
Update Axis1                             // Double buffered registers are copied into
                                         // the active registers, thereby initiating the move
```

2.2.2 Example C-Motion Commands

Here is the same sequence as it would look in actual C-Motion calls:

```
PMDSetProfileMode(&hAxis1, PMDProfileTrapezoidal);
PMDSetPosition(&hAxis1, 12345);
PMDSetVelocity(&hAxis1, 223344);
PMDSetAcceleration(&hAxis1, 1000);
PMDSetDeceleration(&hAxis1, 2000);
PMDUpdate(&hAxis1);
```

In this example hAxis1 is a handle to a structure known as a PMDAxisHandle. There are other commands that address different card resources. For example C-Motion functions that address the ***Device*** resource are prefaced with PMDDevice, and C-Motion functions that address the ***Peripheral*** resource are prefaced with PMDPeriph. Each of these command types take a handle to a variable with that structure. For example commands prefaced with PMDDevice take a handle to a structure known as a PMDDeviceHandle, and commands prefaced with PMDPeriph take a handle to a structure known as a PMDPeripheralHandle.

There are many examples of C-Motion structures, and in general they are used to make accessing the Prodigy/CME Machine-Controller cards simpler and more flexible. In particular, by developing code with C-Motion, it is very easy to change the physical location of a PMD axis or other resource type such as a ***Peripheral*** without any changes to the developed C-Motion code sequences.

Two manuals describe how the Magellan Motion Processor operates and how it is programmed: the *Magellan Motion Processor User's Guide*, and the *Magellan Motion Processor Programmer's Command Reference.* These documents also describe Visual Basic Support, and C-Motion, which are the software libraries that are used to send commands to the Magellan chip and exercise its many functions.

The manual that describes the PRP C-Motion system is the *PMD Resource Access Protocol Programmer's Reference.*

To simplify the presentation of command names, most portions of this manual will provide C-Motion commands without the "PMD" preface. The actual commands however should include the PMD preface. For example the C-Motion command given in the manual as PeripheralRead is actually called PMDPeripheralRead.

2.2.3 Quadrature Encoder Input

Each axis provides differential or single-ended input of A & B quadrature inputs along with an Index signal. These signals provide position feedback to the motion controller which is used to track motor position. For DC brush and brushless DC motors, they are required for proper operation. For step motors, they are optional.

The encoder-processing circuitry provides a multi-stage digital filter of the ***QuadA, QuadB***, and ***Index*** signals for each axis. This provides additional protection against erroneous noise spikes, thus improving reliability and motion integrity.

2.2.3.1 C-Motion Commands

There are numerous Magellan C-Motion commands that relate to the encoder feedback and index position, including commands that retrieve, capture, compare, set, or otherwise utilize the current encoder position. Refer to the *Magellan Motion Processor User's Guide* for more information.

2.2.3.2 Connections & Associated Signals

These signals are named ***QuadA1+*** through ***QuadB4-*** (16 signals), and ***Index1+*** through ***Index4-*** (8 signals), and are all located on the Axis Feedback connectors, as are the grounds that should be used with these signals.

A +5V output on each Axis Feedback connector is provided as a convenience for the encoder to power its internal circuitry. As was the case for the quadrature input signals, one or more of the digital grounds must also be connected to access the +5V.

The quadrature and index signals can be connected in one of two ways. Single-ended means that only one wire per signal is used, while differential means two wires encode each signal (labeled + and -). Differential transmission is generally recommended for the highest level of reliability, because it provides greater noise immunity than a single-ended connection scheme.

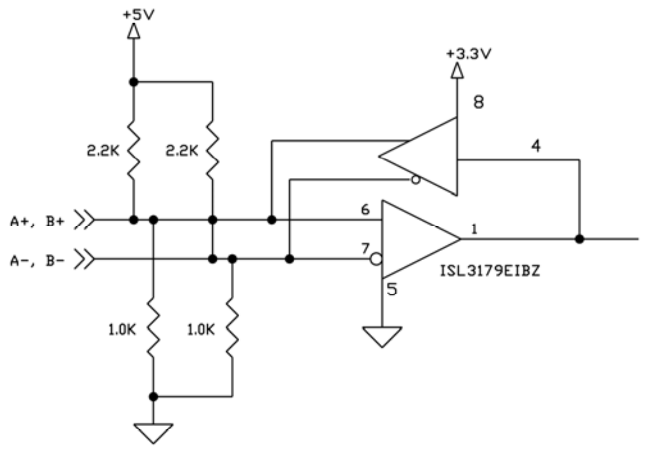
If single-ended connections are used, only the + wire is connected, and the - wire should be left floating. For example, in connecting to the A quadrature input, ***QuadA1+*** connects to the encoder's quadrature A signal, and ***QuadA1-*** remains floating. If differential connections are used, both the + and - signals are used. Differential or single-ended termination must be selected through resistor pack installation. See the table in Section 1.9, "Preparing the Card for Installation" for details.

When using the system with differential connections, the polarity of the differential signal can be reversed by swapping the + and - connections. This may be useful for altering the motor and/or encoder direction, however, this same function can also be accomplished through commands to the Prodigy/CME Machine-Controller card. See the *Magellan Motion Processor User's Guide* for more information.

See Chapter 3, "Electrical Reference" for a complete description of the pinout connections to and from the card.

2.2.3.3 Electrical Interfacing

All of the QuadA, QuadB and Index inputs utilize the following on-card circuitry to process these signals:



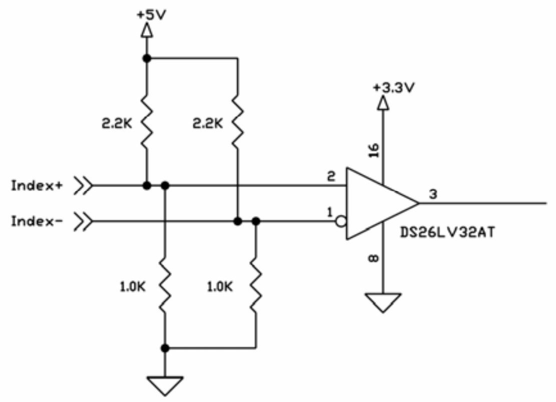


Figure 2-2: Encoder Signal Schematic QuadA/B and Index

2.2.4 SSI (Synchronous Serial Interface) Encoder Input

In addition to incremental quadrature feedback, the machine controller card supports Absolute SSI format encoder feedback for one or more connected axes. The Absolute SSI protocol is used to access high resolution absolute encoders that support this format.

Figure 2-3 shows a typical connection scheme. The machine controller serves as the SSI master and generates the clock signals. The encoder provides its current position at constant, regular intervals based on the SSI communication settings.

On-card logic processes the SSI protocol, converting the axis position to a format that allows it to be input to the Magellan motion processor, where it is used for servo processing and other encoder-related functions.

There are three SSI interface settings that can be set by the user. They are clock frequency, encoder resolution, and word format. The following table shows the range of settings for these parameters supported by the machine controller card:

Parameter	Range	Default Value	Comments
SSI clock frequency	500 kHz - 4.0 MHz	500 kHz	The user should set the clock frequency at the highest rate supported by the SSI encoder and cable length they will be using. Check the encoder specification for details.
SSI encoder resolution	8 bits - 31 bits	25 bits	This should be set to match the encoder word length generated by the SSI encoder being used.
SSI encoder word format	binary or Gray	binary	This should be set to match the encoder format generated by the SSI encoder being used.
SSI enable	enabled or disabled	disabled	To enable SSI operation this field must be set to enable.
SSI clock direction	input or output	input	For normal SSI interfacing the clock will be set for output. For setups where one SSI encoder will be input to multiple cards, the non-clock generating cards should be set to input.

The SSI clock frequency and SSI encoder resolution parameters are controlled via dedicated PIO registers for each axis. See Section 2.1.2, “Peripheral I/O Space” for the complete PIO register address map. The SSI encoder word format, SSI enable, and SSI clock direction parameters are encoded in a single PIO register called SSI configuration. There is one SSI configuration register per axis.

The follow table shows the bit mapping of the SSI configuration register:

Bit no.	Field name	Description
0	SSI Clock direction	A 1 in this field sets the clock signal generation for output. A 0 in this fields sets the clock signals to be input from an external source.
1	SSI Enable	A 1 in this field enables SSI operation. A 0 disables SSI operation.
2-5	SSI Encoder word format	A 0 value in this four-bit field sets the encoder format to binary. A 1 value in this four-bit fields sets the encoder format to gray code.
6-15	reserved	Reserved, may contain zeroes or ones.

Each axis allows independent control of all of these parameters, and all controllable parameters are located in the Peripheral I/O space. The table in Section 2.1.2, “Peripheral I/O Space,” provides the addresses and format of each control register.

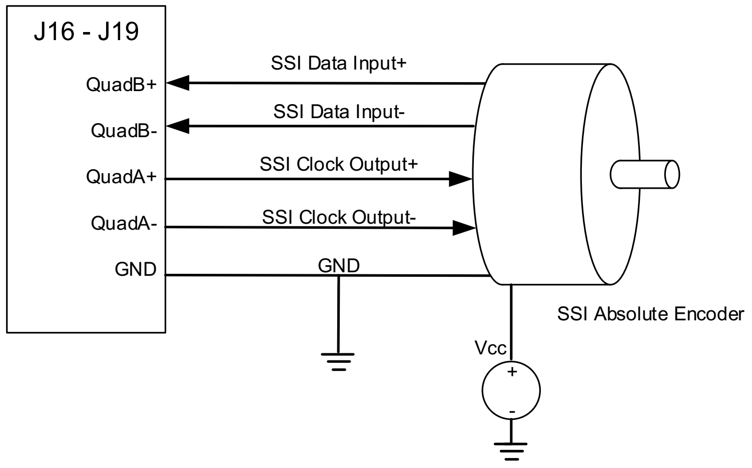


Figure 2-3: SSI with Controller as Clock Master

2.2.4.1 Connecting To Multiple Cards

There may be situations where it is desirable for a single SSI encoder to drive encoder inputs on multiple machine controller cards. The most common reason for this is that an SSI encoder will serve as the electronic gear 'master' input for multiple cards.

This configuration is shown in Figure 2-4.

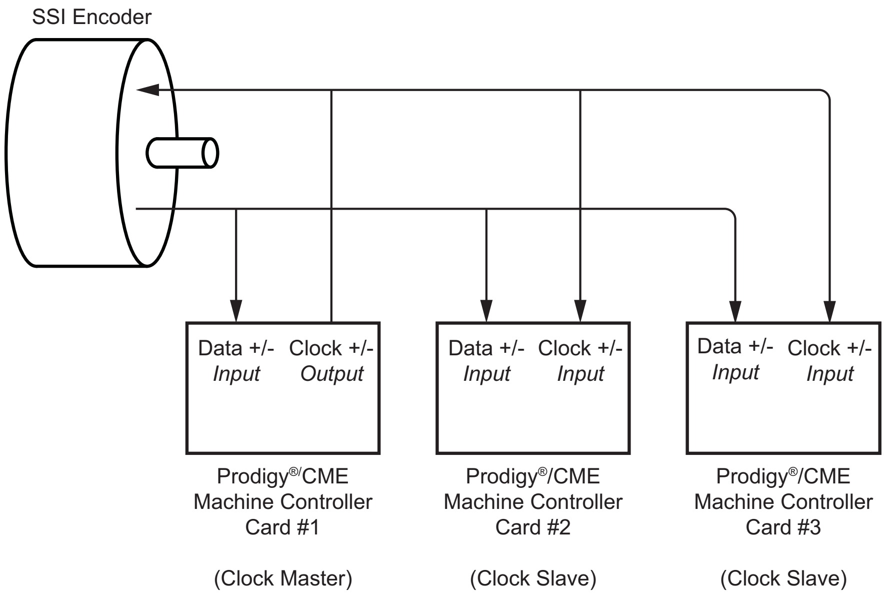


Figure 2-4: Connecting Multiple Cards

To support this configuration the machine controller allows the clock signals for a given SSI axis to be input, rather than output, by the SSI control logic. This is controlled via the SSI clock direction field located in the SSI configuration register.

2.2.4.2 SSI Operation

To enable SSI encoder operation for a particular axis the various control parameters such as word size and clock frequency should be programmed according to the SSI encoder specifications. Then the SSI Enable, SSI clock direction, and SSI encoder word format fields should be set via a single write to the SSI configuration register.

Following this, the Magellan motion processor should be set for 32-bit parallel mode to enable proper data transfer from the card's SSI decode logic.

2.2.4.3 C-Motion Commands

There are several commands needed to set up a particular axis to process SSI encoder information. The Magellan IC encoder input source must be changed from the default value of incremental quadrature to parallel 32-bit word. To change the encoder input source the C-Motion command SetEncoderSource with parallel 32-bit word format selected is used. To read back the current value the command GetEncoderSource is used.

In addition, if the motor encoder is mounted on a motor that can spin multiple times, a modulus should be specified so that the Magellan IC can correctly track the encoder location as it wraps from maximum to smallest value or vice versa. This is accomplished using the C-Motion SetEncoderModulus command. To read the current value back the command GetEncoderModulus is used.

Refer to the *Magellan Motion Processor User's Guide* for more information on both of these commands.

To set or retrieve the value of the machine controller's PIO setup registers the user must first open a Peripheral of type PIO. This is accomplished using the C-Motion command PeriphOpenPIO.

Using the returned *Peripheral* handle, the user can write to the desired SSI registers using the C-Motion command PeriphWrite. To read back the current value of a PIO register the command PeriphRead is used.

For detailed PIO address register information see Section 2.1.2, "Peripheral I/O Space".

2.2.4.4 Connections & Associated Signals

The signals that support SSI are shared with the quadrature signals. Therefore when an SSI encoder is connected it is not possible to have incremental quadrature connections also input to the card. All of the SSI signals are located on the Axis Feedback Axis connectors.

The following tables shows how to connect SSI encoders:

SSI Absolute Signal name	Prodigy/CME Machine-Controller Signal name
Data+	QuadB+
Data-	QuadB-
Clock+	QuadA+
Clock-	QuadA-

In addition to these signals a digital ground should be connected. A +5V output on each Axis Feedback connector is also available, and can be used to power the SSI encoder circuitry.

See Chapter 3, "Electrical Reference" for a complete description of the pinout connections to and from the card.

2.2.4.5 Electrical Interfacing

Refer to Figure 2-2 for the QuadA, QuadB and Index input on-card circuitry.

2.2.5 Home, Limits, Hall Sensors

These signals are conditioned by the card and input directly to the Magellan Motion Processor. The *Magellan Motion Processor User's Guide* explains the functions provided in connection with these various signals. Most of the signals are optional, and are connected depending on the nature of the application. Hall sensors are used only when connecting to brushless DC motors.

2.2.5.1 C-Motion Commands

There are numerous Magellan C-Motion commands related to processing home or limit switches as well as Hall sensor input signals. Refer to the *Magellan Motion Processor User's Guide* for complete information.

2.2.5.2 Connections & Associated Signals

These signals are named *Home1-4, PosLim1-4* (positive direction limit input), *NegLim1-4* (negative direction limit input), and *Hall1A-4C* (12 signals in all). They are all located on the Axis Feedback connectors, as are the ground signals that should be used in conjunction with these inputs.

These signals are single-ended digital inputs to the card. One or more of the digital grounds must be connected. The input signals are pulled up through 4.7k Ohm resistors to 5V.

See Chapter 3, "Electrical Reference," for a complete description of the pin out connections to and from the card.

2.2.5.3 Electrical Interfacing

All of the home, limits, and Hall sensor inputs utilize the following on-card circuitry to process these signals:

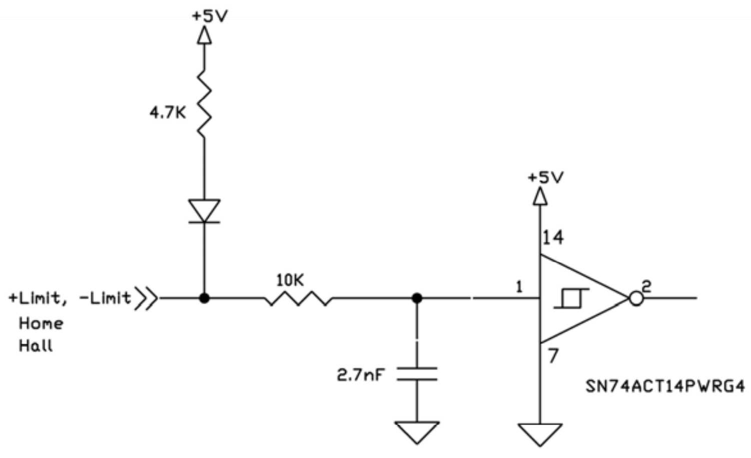


Figure 2-5: Home, Limits, and Hall Sensor Signal Schematic

2.2.6 AxisIn, AxisOut Signals

These signals are input to, or output by, the Magellan motion processor and facilitate coordination of Magellan motion sequences with external hardware. These signals are optional, and are connected depending on the nature of the application.

See the *Magellan Motion Processor User's Guide* for more information on how these signal functions are programmed.

2.2.6.1 C-Motion Commands

There are a number of Magellan C-Motion commands related to processing or generating the AxisIn and AxisOut signals. Refer to the *Magellan Motion Processor User's Guide* for complete information.

2.2.6.2 Connections & Associated Signals

These signals are named *AxisIn1-4*, and *AxisOut1-4*, and are located on the Amplifier IO connector. These signals are single-ended digital inputs to the card.

To function properly, one or more of the digital grounds must be connected.

The input signals are pulled up through 4.7k Ohm resistors to 5V. The default power-up value for all AxisOut signals is high.

See Chapter 3, "Electrical Reference" for a complete description of the pinout connections to and from the card.

2.2.6.3 Electrical Interfacing

AxisIn signal processing is identical to home, limits, and Hall sensor input and is shown in Figure 2-6.

Figure 2-6: AxisIn Signal Schematic

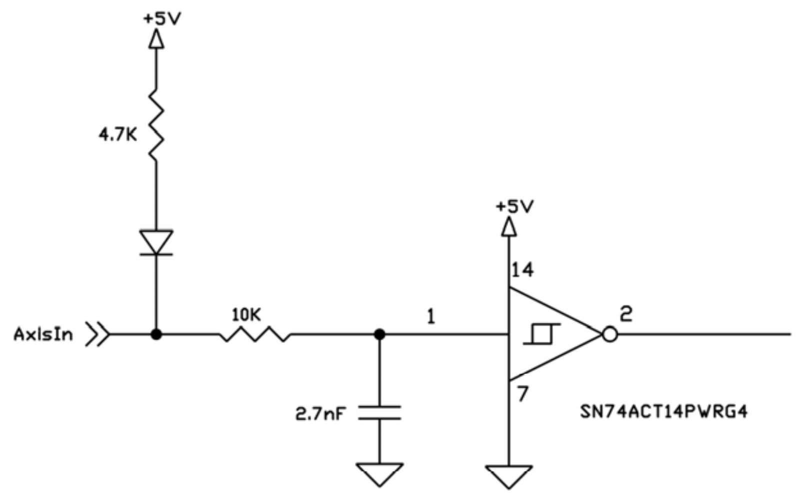
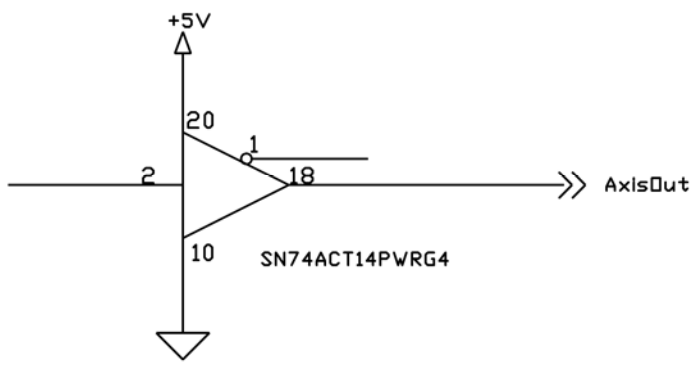


Figure 2-7: AxisOut Signal Schematic

The *AxisOut* signal is generated using the following circuitry:



2.2.7 Atlas Interfacing

The Magellan motion processor communicates to installed Atlas amplifiers via an SPI (Serial Peripheral Interface) bus. The format and protocol of this bus is detailed in the *Atlas Digital Amplifier Complete Technical Reference*, however all Magellan/Atlas communications are handled automatically without need for user action.

Many Atlas control parameters are user-settable, and these parameters are accessed through the Magellan motion processor. The Magellan IC provides a single, seamless, interface to the user by determining whether commands sent to it should be routed to the Atlas amplifier or processed by the Magellan IC. See Section 2.5, "Atlas Amplifier Functions" for more information on overall Atlas function. See the *Magellan Motion Processor User's Guide* for information on the Magellan/Atlas control architecture.

2.2.7.1 C-Motion Commands

There are numerous Magellan C-Motion commands that relate to Atlas amplifier operation. Refer to the *Magellan Motion Processor User's Guide* for complete information.

2.2.7.2 Connections & Associated Signals

There are no connections or signals associated with this function. All of the signals interconnecting the Magellan motion processor and Atlas digital amplifiers are internal to the card. For information on connections associated with the Atlas amplifiers, see Section 2.5, "Atlas Amplifier Functions."

2.2.8 Analog Motor Command Output

The Magellan motion processor can output a desired voltage or torque command on analog output signals. There are eight channels in total, with each channel consisting of 16-bit +/- 10V signals (16 signals in all). When driven by the Magellan motion processor these analog outputs carry the desired voltage or torque for a given axis, and are designed to interface to an off-card motor amplifier.

Analog motor outputs are useful when on-card Atlases are not being used as the amplifier for a particular axis. This may be the case when the motor requires a higher voltage or current power range than the Atlases provide, or when the amplifier must have particular motor-specific characteristics not provided by the Atlas amplifiers.

To interface to a DC brush motor amplifier, or to a brushless DC motor which will be commutated externally, one analog output channel is used per axis. Brushless DC motors that are commutated by the Magellan motion processor use two channels per axis, and interface to amplifiers that support this two-phase input format.

The following table shows how the analog outputs should be connected for each of these motor types:.

Motor Type	Axis #	Channel
DC brush (or externally commutated brushless DC motors)	1	AnalogOut1
	2	AnalogOut2
	3	AnalogOut3
	4	AnalogOut4
Brushless DC (two-phase signal generation)	1	AnalogOut1 (phase A), AnalogOut5 (phase B)
	2	AnalogOut2 (phase A), AnalogOut6 (phase B)
	3	AnalogOut3 (phase A), AnalogOut7 (phase B)
	4	AnalogOut4 (phase A), AnalogOut8 (phase B)

2.2.8.1 AnalogOut Enable

To help insure that machine controller and external amplifier power startups do not induce unexpected motion, the machine controller card provides a mechanism to separately enable or disable the analog output channels. The default on power up is disabled, resulting in a zero voltage output at the analog output signals.

> The analog output channels must be enabled by the user before they can output non-zero voltage values.

Note that the machine controller's analog output circuitry can be controlled by two separate sources. One source is the Magellan motion processor as described above. The other source is via the PRP interface. The default is control by the Magellan motion processor. For information on controlling the analog outputs through the PRP interface see Section 2.3.5, "General Purpose Analog Output."

2.2.8.2 C-Motion Commands

Upon powerup the machine controller card will automatically set any axes that do not have an installed Atlas amplifier to analog output mode. Therefore, in most cases it is not necessary to explicitly set a particular axis to analog motor output mode.

If desired, the motor output mode for an axis can be programmed directly. To set an axis for analog output mode the C-Motion command SetOutputMode is used. The value set can be read back using GetOutputMode. See the *Magellan Motion Processor User's Guide* for more information.

> If the Magellan motion processor is instructed to send motor commands via analog output, any connected Atlas amplifier will immediately cease to communicate with the Magellan. Care should therefore be taken when commanding analog output mode for axes that have an installed Atlas amplifier.

2.2.8.3 Connections & Associated Signals

These signals are named AnalogOut1-8+ and AnalogOut-1-8, and are located on the Amplifier IO connector. They are differential analog outputs which vary between -10V and +10V.

For the analog outputs to function correctly, AGND (analog ground) must be connected. There are three analog grounds available, all located on the Amplifier IO connector.

2.2.8.4 Electrical Interfacing

The analog output signals are generated using the following circuitry:

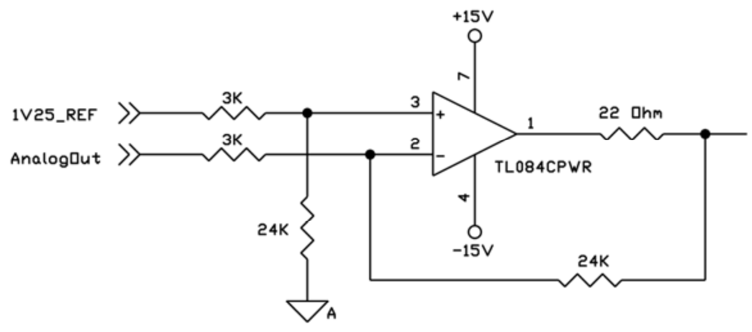


Figure 2-8: Analog Output Signal Schematic

2.2.9 Magellan Watchdog Timer

The machine controller card provides a watchdog function for Magellan I/O transactions. When enabled, the watchdog automatically triggers a Magellan reset if communication to the Magellan should be lost. The most common use of this feature is to allow motion functions to be safely shut down if on-card user application code running on the C-Motion Engine unexpectedly stops sending commands to the Magellan, or if an external controller such as a PC unexpectedly stops sending commands to the Magellan.

See Section 2.2.10, “Magellan Reset,” for a description of the actions and signal changes that occur after a card reset.

The Magellan watchdog functions by requiring a specific value (0x5562) be sent to a specific Magellan I/O address (4) at intervals not exceeding 104 mSec. As long as the watchdog value is written within the 104-mSec interval, no reset will occur and Magellan operations will proceed normally. Once enabled, the watchdog mechanism cannot be stopped until a reset or power cycle occurs.

After powerup or Magellan reset, if no command is sent to the watchdog address, then the watchdog will remain disabled. The watchdog is disabled by default at power-up. When the watchdog timer times out, it also disables itself.

2.2.9.1 C-Motion Commands

The C-Motion command WriteIO is used to specify a Magellan I/O write operation. Proper operation of the watchdog function (such that a timeout is not triggered) requires that a value of 0x5562 be written to the Magellan's I/O address 4 no less often than once every 104 mSec.

2.2.9.2 Connections & Associated Signals

Although as noted above a watchdog timeout may affect various card signals, there are no specific signals associated with this function.

2.2.10 Magellan Reset

Although a reset occurs automatically during power-up, it is sometimes desirable to reset the Magellan Motion Processor explicitly through a user-initiated action. Note that this type of reset is different than a full card reset. See Section 2.6.5, “Card Reset,” for a description of the card reset function.

After a Magellan reset occurs some of the Prodigy/CME Machine-Controller card’s output signals will be driven to known states. These are summarized in the following table:

Signal Name	State
AxisOut 1-4	High
AnalogOut 1-8	Depends on analog output source

Signal Name	State
AmpEnable 1-4	No change

2.2.10.1 C-Motion Commands

The C-Motion command MPDeviceReset causes a reset of the Magellan Motion Processor.

2.2.10.2 Connections & Associated Signals

Although as noted in the table above some card signals may be affected by this command, there are no signal connections specifically associated with this feature.

2.3 I/O Functions

2.3.1 General-Purpose Digital I/O

In addition to signals that directly interface to the Magellan motor processor such as *AxisIn*, *AxisOut*, *Home*, *QuadA*, *QuadB*, *Index*, the machine controller card supports general-purpose digital input and output signals through its General IO connector. There are four digital outputs, four digital inputs, and eight bi-directional I/Os.

These resources along with a few others such as analog input and output are part of the machine controller's 'Peripheral I/O' space, or PIO for short. See Section 2.1.2, “Peripheral I/O Space,” for more information on the machine controller’s PIO address map.

The general purpose digital inputs are TTL-compatible with typical input range of 0–5.5V. The absolute maximum input range is -0.5–7V.

The general purpose digital outputs are 5V TTL-compatible with an output sink / source current of 24mA.

2.3.2 Digital Inputs

There are 12 general purpose digital inputs (4 dedicated inputs, 8 bi-directional). The status of these signals are contained in the lower 12 bits of a C-Motion returned 16-bit word. The following table correlates the signals input with the bits of this returned word.

Signal Name	General IO Connector Pin No.	Bit No.
DigitalIn1	28	0
DigitalIn2	29	1
DigitalIn3	43	2
DigitalIn4	44	3
DigitalIO1	9	4
DigitalIO2	10	5
DigitalIO3	11	6
DigitalIO4	25	7
DigitalIO5	26	8
DigitalIO6	27	9
DigitalIO7	40	10
DigitalIO8	41	11

2.3.3 Digital Outputs

There are 12 general purpose digital outputs (4 dedicated output, 8 bi-directional). These digital output signals are controlled via a write register that holds the desired output signal levels along with a separate register that holds a write mask. Each bit position in the write mask with a value of one will result in the value in the corresponding bit of the write register becoming the commanded digital output value for that signal. Each bit position in the mask with a value of zero will result in the corresponding bit of the write value being ignored.

A separate register can be read to determine the actual output signal commands. This register can not be directly written to. To change the actual output signal commands the write value and write mask registers described above must be used.

For convenience, the following table correlates the signals outputs with the bits of the write register and mask.

Signal Name	General IO Connector Pin No.	Bit No of write register and mask register
DigitalOut1	13	0
DigitalOut2	14	1
DigitalOut3	15	2
DigitalOut4	30	3
DigitalIO1	9	4
DigitalIO2	10	5
DigitalIO3	11	6
DigitalIO4	25	7
DigitalIO5	26	8
DigitalIO6	27	9
DigitalIO7	40	10
DigitalIO8	41	11

2.3.3.1 C-Motion Commands

To set or retrieve the value of the machine controller's general purpose digital I/O the user must first open a Peripheral of type PIO. This is accomplished using the C-Motion command PeriphOpenPIO.

Using the returned *Peripheral* handle, the user can write the 16-bit word value and mask using the C-Motion command PeriphWrite. Note that a single two-word write must be used with the write mask in the low 16-bit word and the write value in the high 16-bit word.

To read back a previously written write value or mask, or to read back the current actual output signals commands, or to read back the current input signal states, the command PeriphRead is used.

PIO space writes to the general purpose digital I/O write value and mask register should always be made via a single call to PeriphWrite.

2.3.3.2 Connections & Associated Signals

The general-purpose digital I/O signals are all located on the General IO connector, J13. One or more of the digital grounds must be connected. Digital inputs are pulled up through 4.7k Ohm resistors to 5V. The power-up default value for all general-purpose digital outputs is low.

See Chapter 3, "Electrical Reference" for a complete description of the pinout connections to and from the card.

2.3.3.3 Electrical Interfacing

The digital inputs are open-collector logic. When an output pin is set to 1 (high), the output transistor is off and the pin can be used as an input. When an output pin is set to 0 (low), the output transistor is on which pulls the signal to ground and the pin cannot be used as an input.

The digital signals are processed using the following circuitry:

Figure 2-9: DigitalIn(1-4) Interface Schematic

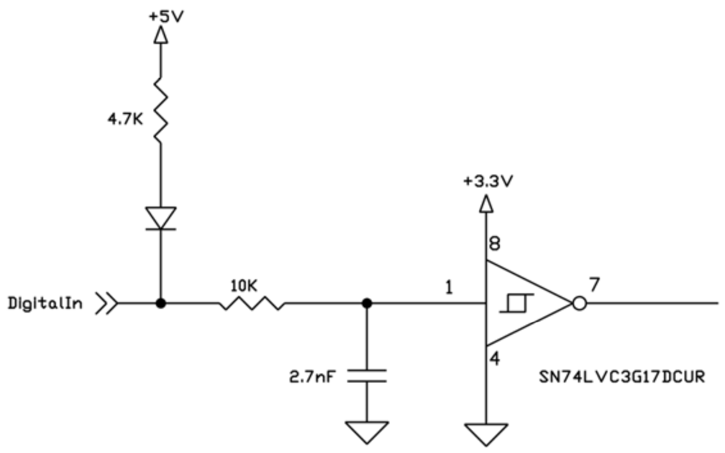
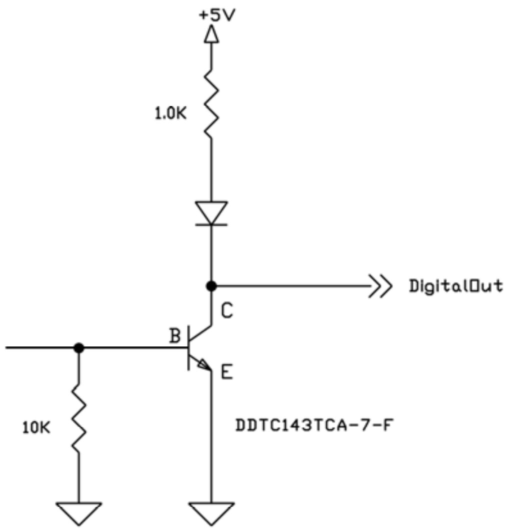


Figure 2-10: DigitalOut(1-4) Interface Schematic



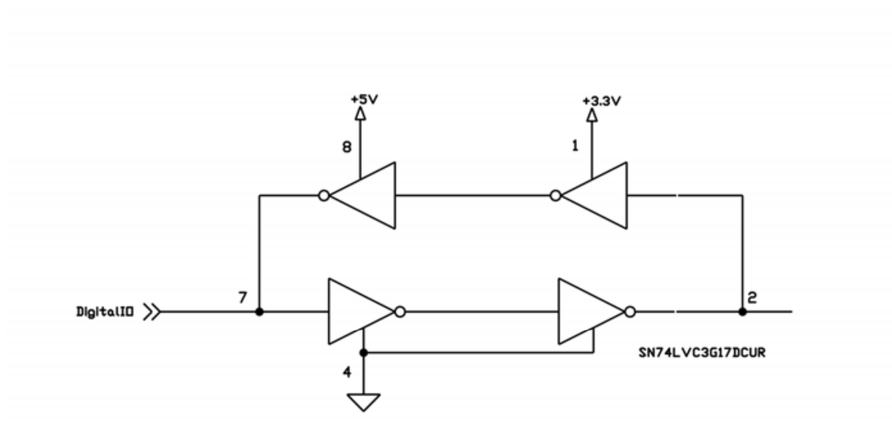


Figure 2-11: DigitalIO(1-8) Interface Schematic

2.3.4 General Purpose Analog Input

The machine controller card supports eight general purpose analog inputs, each consisting of+/- 10V differential analog inputs. The A/D sampling resolution is 16 bits, and the range of the returned value is -32,767 to +32,767 where -32,767 = -10V, and +32,767 = +10V.

The Prodigy/CME Machine-Controller receives both single-ended and differential analog inputs as shown in the following table and Figure 2-12. When used in single ended configuration, AnalogIn- should be connected to ground of the transmitter. When used in differential configuration, AnalogIn+ and AnalogIn- are connected to the differential output of the transmitter. In addition, it is recommended to connect the ground of the transmitter to AGND of the card. Please refer to Figure 2-12

Analog Input Characteristics	Value/Range
Input signal voltage range	-10V to +10V
Resolution	16 bits
Maximum recommended input signal frequency	2 Khz
Throughput	40 kilo samples per second
Integral non linearity error, typical	+/- 1 LSB
Integral non linearity, maximum	+/- 3LSB
Overvoltage protection	-30V to +30V
Common-mode range	-12V to +15V
Typical differential impedance	28 Kohms
Common-mode rejection ratio, minimum:	86 db

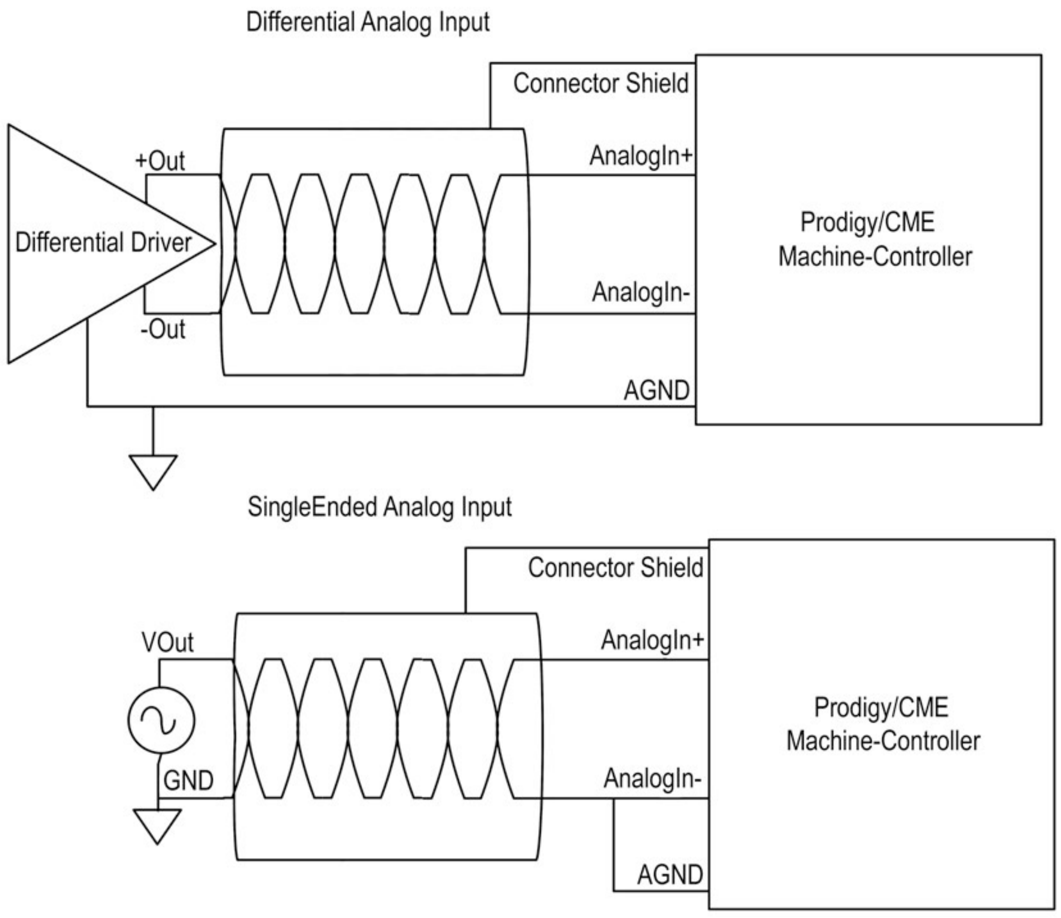
To determine the numerical value that will be read by the Prodigy/CME Machine-Controller card given a specific voltage at the input pins, the following formula is used:

ReadValue = AnalogVoltage*32,767/10.0V

Conversely, given a read value, the voltage at the connection is calculated as:

AnalogVoltage = ReadValue*10.0V/ 32,767

Figure 2-12: Analog Inputs Simplified Diagrams



2.3.4.1 C-Motion Commands

To read the machine controller's analog input channels the user must first open a Peripheral of type PIO. This is accomplished using the C-Motion command PeriphOpenPIO.

Using the returned *Peripheral* handle, the user can read one or more 16-bit word values representing the current analog voltage using the PeriphRead command.

2.3.4.2 Connections & Associated Signals

The general-purpose analog inputs are differential analog inputs located on the General IO connector, J13. To function properly one or more of the analog grounds must be connected.

See Chapter 3, “Electrical Reference” for a complete description of the pinout connections to and from the card.

2.3.4.3 Electrical Interfacing

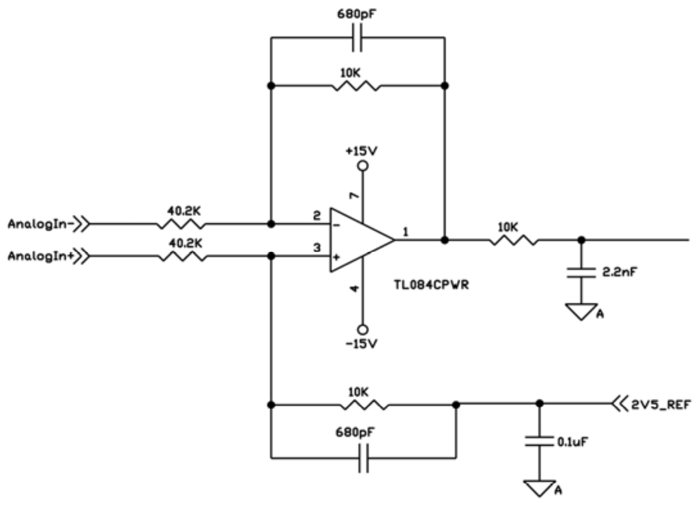


Figure 2-13: Analog Input Interface

2.3.5 General Purpose Analog Output

There are 8 general purpose analog outputs, each consisting of a 16-bit +/- 10V differential analog output. The range of the commanded analog output word is 0-65,535 where 0 = -10V, and 65,535 = +10V.

The analog outputs can be controlled by two separate sources. One source is the Magellan motion processor which uses them to output desired motor torque commands. The other source is user control via the PRP interface. Each of the 8 channels can be individually directed to be controlled by the Magellan or by the user. The default analog output source of all analog output channels is the Magellan motion processor.

The AnalogOut source is controlled via axis-specific registers in the PIO space. See Section 2.1.2, “Peripheral I/O Space,” for the detailed location and format of these registers.

Analog Output Characteristics	Value/Range
Output signal voltage range	-10V to +10V
Resolution	16 bits
Integral non linearity error, typical	+/- 2 LSB
Integral non linearity, maximum	+/- 1 LSB
DC output impedance, typical	0.5 ohm
Short circuit current	30mA

To determine the voltage that will be output by the Prodigy/CME Machine-Controller card given a specific commanded output value the following formula is used:

AnalogVoltage = WriteValue*10.0V/65,535

Conversely, given a desired voltage the value that should written to generate this voltage is calculated as:

WriteValue = AnalogVoltage*65,535/10.0V

2.3.5.1 AnalogOut Enable

The Prodigy/CME Machine-Controller cards allow the AnalogOut signals, described in Section 2.2.8, “Analog Motor Command Output,” to be shunted to 0 volts for safety purposes (disabled), or to be actively controlled (enabled). The

AnalogOut enable/disable mechanism is 'global,' meaning it affects all AnalogOut channels. It is not possible to selectively enable/disable analog output by axis or channel.

The power up default value for the AnalogOut enable function is disabled. In addition, the AnalogOut enable function is disabled upon a card reset, or via the external Reset signal. See Section 2.6.5, "Card Reset," for more information.

The analog output channels must be enabled by the user before they can output non-zero voltage values. See Section 2.2.8.1, "AnalogOut Enable" for information on how to control the AnalogOut signal enable/disable.

2.3.5.2 C-Motion Commands

To set the machine controller's analog output channels the user must first open a Peripheral of type PIO. This is accomplished using the C-Motion command PeriphOpenPIO.

Using the returned *Peripheral* handle, the user can write one or more 16-bit word values representing the desired analog voltage command, the AnalogOut source, or AnalogOut enable/disable status using the PeriphWrite command. The written values can be read back using PeriphRead.

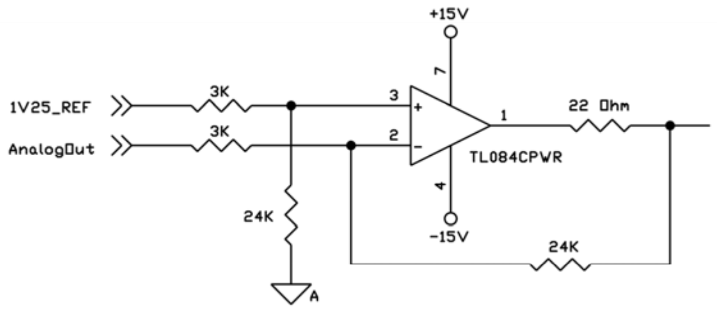
2.3.5.3 Connections & Associated Signals

The general-purpose analog outputs are direct differential analog outputs located on the Amplifier IO connector, J14. One or more of the analog grounds must be connected.

See Chapter 3, "Electrical Reference," for a complete description of the pinout connections to and from the card.

2.3.5.4 Electrical Interfacing

Figure 2-14: Analog Output Interface



2.4 Communications Functions

The Prodigy/CME Machine-Controller card provides three overall communication interfaces, Serial, CANbus, and Ethernet.

Basic access to either the Serial, CANbus, or Ethernet port is accomplished by sending a C-Motion command with the detailed connection parameters that will be used. A ***Peripheral*** is used to access the connection once it is established.

For example to create an Ethernet TCP connection, the IP Address and port number is provided. If the connection is successfully established, a ***peripheralID*** is generated and thereafter used as the reference for any future communications through that connection.

In the subsequent sections the operational characteristics of the Serial (Serial1 & Serial2), CANbus, and Ethernet network ports will be detailed.

2.4.1 Serial1 & Serial2

Machine-Controller cards provide synchronous serial communications in either RS232 or RS485 mode. Access to the serial port controller is managed using peripheral connections.

In RS232 mode two serial ports are supported, referred to as Serial1, and Serial2. While some applications will not need to use two serial ports, the second port may be useful during C-Motion application code debugging, or to communicate with various serial devices connected to the machine. In RS485 mode, a single serial port is supported, referred to as Serial1. Also in RS485 mode, the serial port may be operated in either half duplex or full duplex mode.

Pin #1 of the J23 Serial Connector selects whether RS232 or RS485 communications mode is used. If left open (the default condition), the card operates the serial port in RS232 mode. If closed (tied to ground) the serial port is operated in RS485 mode. Note that a change in the status of this pin will not properly take effect until after a power on or reset of the card.

Both Serial Port 1 and Serial Port 2 can be operated at various communication settings as shown in the following table. All settable serial port parameters can be programmed separately for Serial 1 and Serial 2. For RS232 communications each serial controller only allows one peripheral to be established at a time. For example, if a peripheral is opened to establish RS232 communications via Serial 1, another peripheral may be opened to establish RS232 communication via Serial 2. However if a third peripheral is then opened to establish a new connection with Serial 1, the original Serial 1 peripheral will automatically be closed.

Settings & default values for Serial1 and Serial2:

Parameter	Range	Serial1 Default	Serial2 Default
Baud rate	1,200 to 460,800	57,600	115,200
Parity	none, even, odd	none	none
# Data bits	5, 6, 7, 8	8	8
# stop bits	1, 2	1	1

After a reset or at power-up the card retrieves default information for the serial ports from the on-card non-volatile RAM. To simplify startup these default values can be changed by the user. See Section 2.6.6, “Setting Card Defaults,” for information.

2.4.1.1 C-Motion Commands

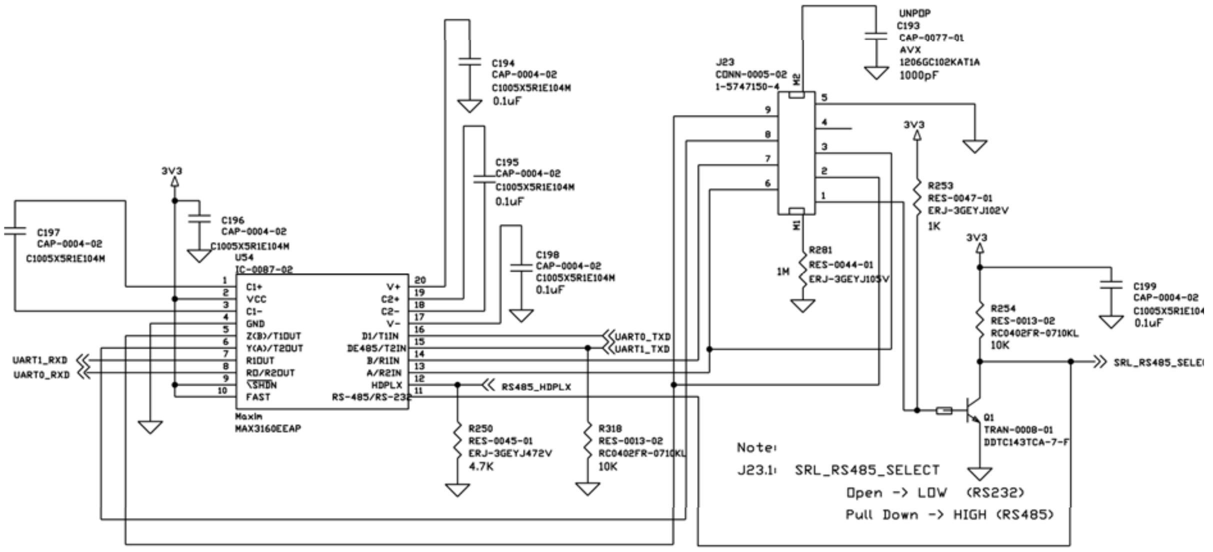
To create a serial port peripheral connection the above parameters are specified in the C-Motion command **PeriphOpenCom**. Messages to and from the Serial port are transmitted via the **PeriphSend** and **PeriphReceive** commands. Whenever a new serial port peripheral is opened its previous function is canceled. By default Serial1 listens for PRP communications and Serial2 is the console port.

For detailed information on PRP action formats and function, refer to the *PMD Resource Access Protocol Programmer's Reference.*

2.4.1.2 Connections & Associated Signals

J23 comprises a special 9-pin connector used to connect one or both serial ports. See Section 3.2.8, “Serial Connector,” for a detailed signal description of the Serial connector.

2.4.1.3 Electrical Interfacing



U54
IC-0087-02
Maxim
MAX3160EEAP
C1+
VCC
C1-
GND
Z(B)/T1OUT
Y(A)/T2OUT
R1OUT
RO/R2OUT
SHDN
FAST
V+
C2+
C2-
V-
DI/T1IN
DE485/T2IN
B/R1IN
A/R2IN
HDPLX
RS-485/RS-232
3V3
C194 CAP-0004-02 C1005X5R1E104M 0.1uF
C195 CAP-0004-02 C1005X5R1E104M 0.1uF
C196 CAP-0004-02 C1005X5R1E104M
C197 CAP-0004-02 C1005X5R1E104M
C198 CAP-0004-02 C1005X5R1E104M 0.1uF
C199 CAP-0004-02 C1005X5R1E104 0.1uF
UNPOP C193 CAP-0077-01 AVX 1206GC102KAT1A 1000pF
J23 CONN-0005-02 1-5747150-4
R281 RES-0044-01 ERJ-3GEYJ105V 1M
R253 RES-0047-01 ERJ-3GEYJ102V 1K
R254 RES-0013-02 RC0402FR-0710KL 10K
R250 RES-0045-01 ERJ-3GEYJ472V 4.7K
R318 RES-0013-02 RC0402FR-0710KL 10K
Q1 TRAN-0008-01 DDTC143TCA-7-F
UART1_RXD
UART0_RXD
UART0_TXD
UART1_TXD
RS485_HDPLX
SRL_RS485_SELE
Note:
J23.1: SRL_RS485_SELECT
Open -> LOW (RS232)
Pull Down -> HIGH (RS485)

Figure 2-15: Serial Interface Schematic

2.4.2 CANbus Communications

The Prodigy/CME Machine-Controller card provides a general purpose CANbus port compatible with the CAN2.0B standard which may be operated at various communication rates from 10,000 to 1,000,000 bps (bits per second). In addition, each CANbus device is assigned two CAN identifiers (also called addresses); one for transmission of messages, and one for reception of messages.

The following table summarizes this information along with the factory defaults for these values:

Parameter	Range	Default
Baud rate	10,000 to 1,000,000 bps	1,000,000
Host send address	0 - 0x800	0x580
Host receive address	0 - 0x800	0x600

After a reset or at power-up the card retrieves default information for the CANbus port from the card's non-volatile RAM. See Section 2.6.6, "Setting Card Defaults," for more information on changing these default values.

2.4.2.1 C-Motion Commands

To create a CANbus peripheral connection, the above parameters are specified in the C-Motion command PeriphOpenCAN. Messages to and from the CANbus port are transmitted via the PeriphSend and PeriphReceive commands.

2.4.2.2 Connections & Associated Signals

J21 and J22 provide standard RJ-45 connectors to connect to a CANbus network. See Section 3.2.9, "CAN Connectors," for a detailed signal description of the CAN connectors.

2.4.2.3 Electrical Interfacing

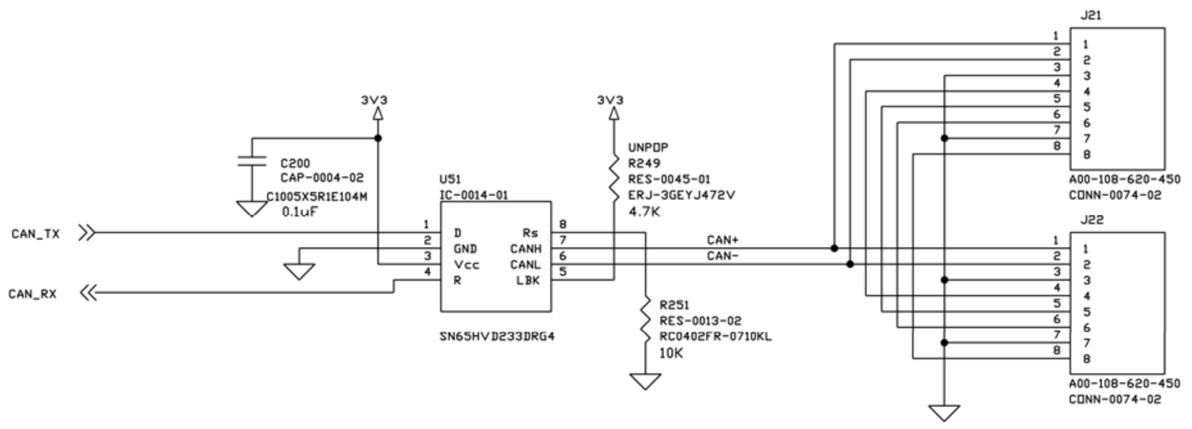


Figure 2-16: CANbus Interface Schematic

2.4.3 Ethernet Communications

The Prodigy/CME Machine-Controller cards support two different Ethernet protocols, TCP (Transmission Control Protocol) and UDP (User Datagram Protocol). TCP is typically used for primary Ethernet communications to the card, while UDP is typically used for non-critical applications such as data logging, or for the Pro-Motion console window. See Section 2.7.6, “Debug Console Window,”for more information on the C-Motion Engine console window.

When used to receive PRP messages the physical node on the Ethernet network controller is assigned a 32-bit IP (Internet Protocol) address, along with a 32-bit netmask and a 32-bit gateway value. The Netmask is used to indicate which IP addresses are local, and the gateway value is used to route non-local addresses. To correctly receive communications from the host controller, a 16-bit identifier known as a port must also be specified. Note that when used as the connection between the host controller and the Prodigy/CME Machine-Controller card, TCP rather than UDP communications are used. To determine what the unused IP addresses are for your Ethernet network, and what values for netmask and gateway to use, you should contact your network administrator.

By convention, the 32 bit values for IP Address, Netmask, and Gateway are shown in Dotted Quad Notation. In this notation each of the four numbers are separated by dots, and denote a decimal value for each byte of the four byte word.

The table below shows the range and default settings for the Ethernet controller of the Prodigy/CME Machine-Controller card:

Parameter	Range	Default
IPaddress	0.0.0.0 –255.255.255.255	192.168.2.2
Netmask	0.0.0.0 –255.255.255.255	255.255.255.0
Gateway	0.0.0.0 –255.255.255.255	0.0.0.0
PRPListenTCPPort	0 - 65,535	40100

Each physical hardware device on an Ethernet network is assigned one IP address, however, a given IP address can have multiple ports. This is useful because it allows user application code running on the C-Motion Engine to open up peripheral connections using port numbers other than the PRP communications port (which has a default value of 40100), thereby allowing PRP messages and application-specific data in any format to co-exist on the same Ethernet IP node.

After a reset or at power-up, the card retrieves default information for the Prodigy/CME Machine-Controller Ethernet port from the card's non-volatile RAM. To change these default values see Section 2.6.6, "Setting Card Defaults."

2.4.3.1 TCP Connection Keep-Alive

All prodigy/CME card TCP connections, including the PRP communications port, use a 'keep-alive' mechanism to detect whether a connection is still valid.

The keep-alive mechanism is a standard part of the TCP protocol specification, and is useful for preventing the prodigy/CME card from leaving connections open if the host has not properly closed a connection. This may occur, for example, if the host has been physically disconnected, or otherwise stops functioning.

The default keep-alive parameters for the Prodigy/CME cards are:

Parameter	Range	Default
idle time	0-65,535 seconds	60 seconds
interval time	not settable	30 seconds
retry count	not settable	2

The *idle time* is the amount of time after the last message on the port that must occur for a 'keep-alive' message to be sent. When sent, the keep-alive message requests the host connection to acknowledge that it is still functioning properly. If it provides this acknowledgment, the *idle time* counter is reset to 0. If it does not, a second 'keep-alive' message will be set after the *interval time*, and this will be repeated a total of *retry count* number of times. If the host ultimately does not correctly respond, the Prodigy/CME connection will automatically be closed.

Note that all of these functions are handled automatically by the Prodigy/CME card's TCP processing system, and should in turn also automatically be handled by the host's TCP system. No user action is required to initiate or monitor these automatic TCP 'keep-alive' messages.

See Section 2.6.6, "Setting Card Defaults," for information on how to change machine controller default values including the keep-alive parameters described above.

2.4.3.2 C-Motion Commands

To create an Ethernet/TCP or Ethernet/UDP peripheral conversation the IP address and port are specified in the C-Motion commands PeriphOpenTCP or PeriphOpenUDP, respectively. To transfer messages via this peripheral connection the PeriphSend and PeriphReceive commands are used

For detailed information on PRP action formats and function, refer to the *PMD Resource Access Protocol Programmer's Reference.*

2.4.3.3 Connections & Associated Signals

The Ethernet connector is a standard RJ-45 connector and is located at J24.

2.5 Atlas Amplifier Functions

2.5.1 Atlas Digital Amplifier Overview

Atlas Digital Amplifiers are single-axis amplifiers that provide high performance torque control of brushless DC, step motor, and DC brush motors. They accept digital torque commands from an external source and are used directly for motor torque control applications, or in conjunction with higher level controllers for velocity or positioning applications. Their very compact size and high power output make them an ideal solution for single-card machine controllers that require high performance in a small envelope.

Atlas digital amplifiers provide many advanced control features including user-programmable gain parameters, performance trace, field oriented control, and I^2t current management. Atlas amplifiers are powered from a single supply voltage, and provide automatic protection from overcurrent, undervoltage, overvoltage, overtemperature, and short circuit faults.

The Atlas family has been designed to work seamlessly with PMD's Magellan family of motion processors. Communication to/from Atlas amplifiers is via SPI (Serial Peripheral Interface) using a simple, packet-oriented protocol. For step motors, in addition to the SPI format a dedicated pulse & direction input mode is provided.

For complete documentation on all aspects of the Atlas Digital Amplifier, refer to *Atlas Digital Amplifier Complete Technical Reference.*

2.5.2 Atlas Drive Ratings

Specifications	DC Brush Motor	Brushless DC Motor	Step Motor
Nominal supply voltage	12-56 VDC	12-56 VDC	12-56 VDC
Continuous current	14.0 ADC	10.0 Arms	9.0 Arms
Peak current (per phase)	25.0 A	25.0 A	25.0 A
Maximum continuous power	670 W	590 W	610 W

2.5.3 Safety Processing Functions

Atlas provides a number of amplifier control features that automatically detect and manage safety-related conditions. In addition, Atlas can signal when various conditions, safety-related or otherwise, occur.

The subsequent sections describe these features.

2.5.3.1 Overcurrent Fault

Atlas supports automatic detection of excessive current output. This fault occurs when the motor, the wiring leading from Atlas, or Atlas unit's power stage becomes short circuited or the requested current is too large to be processed by Atlas.

An overcurrent fault will cause the current loop and power stage modules to be disabled, thereby halting further motor output. To recover from this condition the user should determine the nature of the fault. It is generally desirable to power down Atlas to check connections or otherwise correct the Atlas-attached hardware so that the problem does not occur again.

If the overcurrent condition has been resolved, when restart is attempted Atlas will resume normal operations. If the overcurrent condition has not been resolved, the overcurrent condition will immediately occur again.

Over current faults are serious conditions and warrant the utmost caution before re-enabling amplifier operation. It is the responsibility of the user to determine the cause and corrective action of any electrical fault.

2.5.3.2 Overtemperature Fault

Atlas provides the capability to continually monitor and detect excessive internal temperature conditions. Such a condition may occur if excessive current is requested, if heat sinking of the Atlas unit is inadequate, or if some other problem results in elevated drive temperatures.

To detect this condition a programmable temperature threshold is continuously compared to an internal temperature sensor. If the value read from the internal sensor exceeds the programmed threshold, an overtemperature fault occurs. In addition, a settable overtemperature hysteresis allows the user to ensure that the Atlas temperature drops by a specified number of degrees before allowing drive restart.

The maximum allowed setting for the temperature threshold is 75.0° C, which is also the default value. The maximum allowed value of the hysteresis parameter is 50° C, and the default value is 5° C.

An overtemperature fault will cause the current loop and power stage modules to be disabled, thereby halting further motor output. To recover from this condition the user should determine the nature of the fault. It is generally desirable to power down Atlas to correct the condition.

If the overtemperature condition has been resolved, when restart is attempted Atlas will resume normal operations. If the overtemperature condition has not been resolved, the condition will immediately occur again.

Overtemperature faults indicate that the internal safe limit of the drive temperature range has been exceeded. This potentially serious condition can result from incorrect motor connections, excessive power demands placed on the Atlas amplifier, or inadequate heat sinking. It is the responsibility of the user to operate Atlas within safe limits.

2.5.3.3 Overvoltage Fault

Atlas provides the capability to continually monitor and detect excessive voltages on the incoming voltage supply. Such a condition may occur if there is a fault in the system power supply, if a large back EMF (electromotive force) is generated during motor deceleration, or if some other problem results in an elevated bus voltage.

To detect this condition a programmable bus voltage threshold is continuously compared to the bus voltage sensor. If the value read from the internal sensor exceeds the programmed threshold, an overvoltage fault occurs.

The maximum allowed setting for the overvoltage threshold is 60.0 volts, which is also the default value. The minimum allowed threshold is 10.0 volts.

An overvoltage fault will cause the current loop and power stage modules to be disabled, thereby halting further motor output. To recover from this condition the user should determine the nature of the fault. In most cases it is desirable to power down Atlas to correct the condition.

If the overvoltage condition has been resolved, when restart is attempted Atlas will resume normal operations. If the overvoltage condition has not been resolved, the condition will immediately occur again.

Overvoltage faults indicate that a serious safety condition has occurred. It is the responsibility of the user to operate Atlas within safe limits.

2.5.3.4 Undervoltage Fault

Atlas also provides the capability to sense undervoltage conditions. This value is compared to the value read from the drive DC bus, and if the value read is less than the programmed threshold, an undervoltage fault occurs. The minimum allowed value for this threshold is 10.0 volts, which is also the default value. The maximum allowed value is 56.0 volts.

All other aspects of this feature are the same as for overvoltage sense. Just as for overvoltage conditions, it is the user's responsibility to determine the seriousness of, and appropriate response to, an undervoltage condition.

2.5.3.5 Current Foldback

Current foldback, also known as I^2t foldback, is a general purpose tool to protect the drive output stage or the motor from excessive current.

I^2t current foldback works by integrating, over time, the difference of the square of the actual motor current and the square of a user-settable continuous current limit. When the integrated value reaches a user-settable energy limit, Atlas goes into current foldback. When in this condition, and correctly programmed, Atlas will attempt to clamp the maximum current to the continuous current limit value. Note that the Atlas unit's ability to do so depends on a properly functioning current loop. For more information refer to the *Atlas Digital Amplifier Complete Technical Reference.*

Atlas will stay in foldback until the integrator returns to zero. This is shown in Figure 2-17.

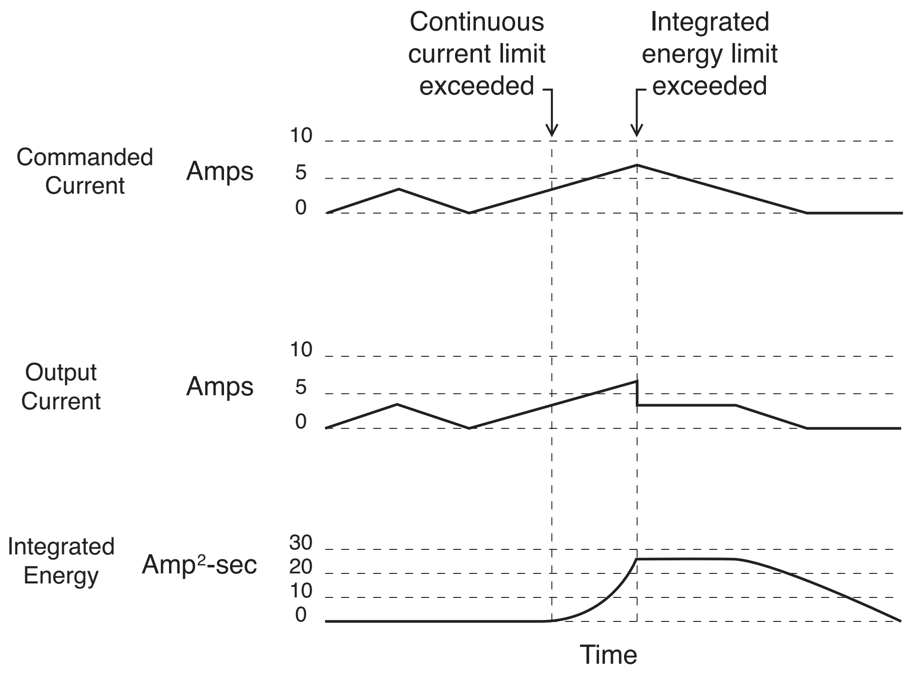


Figure 2-17: Current Foldback Processing Example

Each Atlas amplifier motor type has particular default and maximum allowed values for both the continuous current limit and energy limit. These values are designed to protect the Atlas from excessive heat generation. The table below shows these values.

Motor Type	Continuous Current Limit Default	Continuous Current Limit Maximum	Energy Limit Default	Energy Limit Maximum
Brushless DC	9.0 Amps	10.0 Amps	139 Amp^2Sec	139 Amp^2Sec
DC Brush	12.0 Amps	14.0 Amps	144 Amp^2Sec	144 Amp^2Sec
Step Motor	8.0 Amps	9.0 Amps	149 Amp^2Sec	149 Amp^2Sec

Current foldback, when it occurs, may indicate a serious condition affecting motion stability, smoothness, and performance. It is the responsibility of the user to determine the appropriate response to a current foldback event.

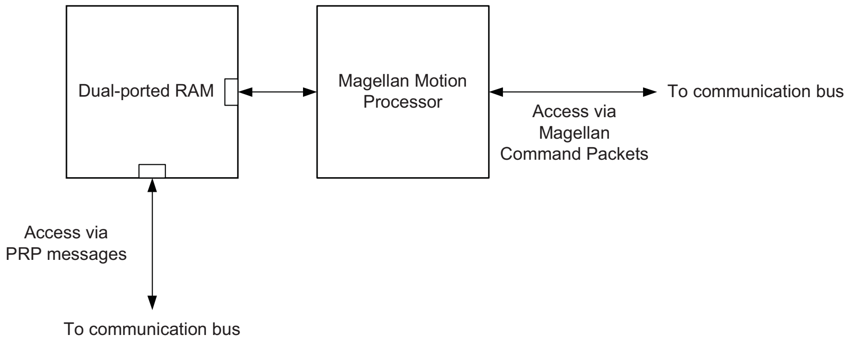
2.6 General Card Functions

There are a number of features and resources on the Prodigy/CME Machine-Controller card that are not directly controlled by the Magellan Motion Processor, or associated with general I/O or the C-Motion engine. The next several sections describe these general card functions.

2.6.1 Dual-Ported RAM (DPRAM)

The Prodigy/CME Machine-Controller card has an on-card dual-ported memory (DPRAM) which has one port interfaced to the Magellan motion processor and the other port interfaced to the card's high speed internal communications bus, allowing two paths of communication. Figure 2-18 shows this configuration.

Figure 2-18: On-card Dual-ported Memory



The dual-ported RAM is most often used to allow various Magellan Motion Processor parameters and registers to be continuously captured and stored to a memory buffer. The captured data may be downloaded to the C-Motion Engine or to an off-card host using the Prodigy/CME Machine-Controller card's serial, CANbus, or Ethernet communication channels.

Magellan data traces are useful for optimizing DC brush and brushless DC servo performance, verifying trajectory behavior, capturing sensor data, or to assist with any type of monitoring where a precise time-based record of the system's behavior is required. For more information on how to set up a trace within the Magellan Motion Processor, see the "Trace Capture" section of the *Magellan Motion Processor User's Guide.*

Beyond trace, the dual ported memory may also be used for general purpose RAM memory storage, particularly by user code located on the C-Motion Engine.

The machine controller card is available with two available memory configurations. The standard dual port memory configuration is 128Kbyte capacity. The enhanced memory option is 468 Kbyte capacity.

The contents of the dual-ported RAM are volatile. They are not saved during power-down of the card.

2.6.1.1 Accessing The Dual-Ported RAM

To access the contents of the dual ported RAM via the Magellan port the Magellan's built in memory buffer commands are used. The Magellan provides a sophisticated command set that lets you set up, monitor, and read from the trace buffer. See the *Magellan Motion Processor User's Guide* for more information on these commands. In this read configuration, the Magellan Motion Processor stores data to the DPRAM autonomously, and the host controller reads the data using the Magellan Motion Processor as well.

An alternate path for reading from, or writing to, the dual ported RAM is via the Prodigy/CME Machine-Controller card's high speed communication bus. In this mode the dual-ported RAM is accessed via the PMD Resource Access Protocol.

2.6.1.2 C-Motion Commands

There are a number of C-Motion commands related to Magellan buffer operations. Refer to the *Magellan Motion Processor User's Guide* for a detailed description of these commands.

To read or write to the dual-ported RAM using the PRP system, a resource address must first be obtained via the MemoryOpen command with memory type DPRAM. To write data to the dual-ported RAM, the C-Motion command MemoryWrite is used. To read the contents of the dual-ported RAM, the C-Motion command MemoryRead is used.

Note that byte-sized memory operations are not supported to the dual-ported RAM.

For complete information on the format and function of these commands, refer to the *PMD Resource Access Protocol Programmer's Reference.*

2.6.1.3 Connections & Associated Signals

There are no signals associated with this function.

2.6.2 Non-volatile Memory

The Prodigy/CME Machine-Controller cards have a general purpose 4,094 byte memory that retains its contents after a card power down or reset. This memory is useful for storing parameters that are set only occasionally and stay with the card, such as machine calibration information.

Accessing the non-volatile memory is accomplished in the same manner as accessing the dual-ported RAM, except that the NVRAM memory type is specified instead of the DPRAM memory type. Addresses are specified from 0 to 1,023. When writing to this memory, a typical write takes 50µSecs, however under certain circumstances it can take much longer, up to several 100mSec. Read speed is the same as for other memory resources, and takes just a few nanoseconds. As for the dual-ported RAM, byte-size memory operations are not supported by the non-volatile memory. The smallest memory unit that can be accessed is 16 bits.

The non-volatile memory can be rewritten a limited number of times. The worst case write limit cycle is 100,000 times for a given memory address, but in typical operation the limit is much higher. As a general guideline, to avoid erase/write cycle limit problems, the non-volatile RAM should not be used for general purpose scratch RAM, and should only be used to store permanent or semi-permanent parameters.

The typical write time to the non-volatile RAM is 50µSec, however it may take as long as several 100 mSec. If other portions of the user application code, or any other PRP-connected device, depends on these values having been written, it is recommended that you ensure that the write operation has been completed by adding code that explicitly checks the value, or by waiting a fixed period of time after the NVRAM write operation.

2.6.2.1 C-Motion Commands

To read or write to the non-volatile RAM a resource address must first be obtained via the MemoryOpen command with memory type NVRAM. To write data to the NVRAM the C-Motion command MemoryWrite is used. To read the contents of the NVRAM the C-Motion command MemoryRead is used.

For complete information on the format and function of these, and other commands, refer to the *PMD Resource Access Protocol Programmer's Reference.*

2.6.2.2 Connections & Associated Signals

There are no signals associated with this function.

2.6.3 Amplifier Enable

The Prodigy/CME Machine-Controller cards provide four digital output signals directly controllable by the user that are intended to be used as amplifier enable signals, one per axis. If not used for this purpose these signals may be used as general purpose outputs.

The four AmpEnable digital output signals are controlled via a write register that holds the desired output signal levels along with a separate register that holds a write mask. This register along with all of the registers used to control the amplifier enable signals is located in the Peripheral I/O space (PIO). See Section 2.1.2, "Peripheral I/O Space," for a complete description of this space. Each bit position in the write mask with a value of 1 will result in the value in the corresponding bit of the write register becoming the commanded digital output value for that signal. Each bit position in the mask with a value of zero will result in the corresponding bit of the write value being ignored.

A separate register can be read to determine the actual output signal commands. This register can not be directly written to. To change the actual output signal commands the write value and write mask registers described above must be used.

For convenience, the following table correlates the signal outputs with the bits of the write register and mask.

Signal Name	Amplifier IO Connector Pin No.	Bit No. of write register and mask register
AmpEnable1	1	0
AmpEnable2	2	1
AmpEnable3	10	2
AmpEnable4	11	3

The power-up default value for all amplifier enable signals is low (disabled).

2.6.3.1 C-Motion Commands

To set or retrieve the value of the machine controller's AmpEnable outputs the user must first open a Peripheral of type PIO. This is accomplished using the C-Motion command PeriphOpenPIO.

Using the returned Peripheral handle, the user can write the 16-bit word value and mask using the C-Motion command PeriphWrite. Note that a single two-word write must be used with the write mask in the low 16-bit word and the write value in the high 16-bit word.

To read back a previously written write value or mask, or to read back the current actual output signals commands, or to read back the current input signal states the command PeriphRead is used.

PIO space writes to the general purpose digital I/O write value and mask register should always be made via a single call to PeriphWrite.

2.6.3.2 Connections & Associated Signals

AmpEnable1-4 are direct digital outputs and appear on the card's Amplifier IO connector. To function properly one or more of the digital grounds must be connected.

See Chapter 3, "Electrical Reference," for a complete description of the pinout connections to and from the card.

2.6.3.3 Electrical Interfacing

The AmpEnable signals are generated using the following circuitry on-card:

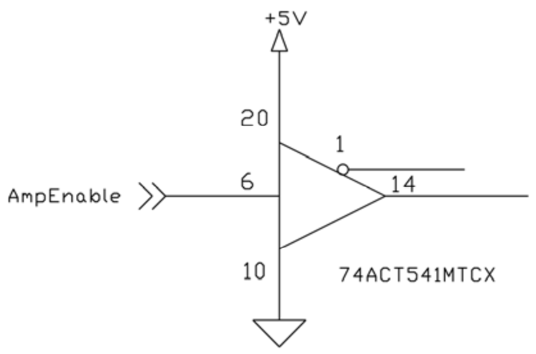


Figure 2-19: AmpEnable Output Interfacing

2.6.4 Reset Monitor

During normal operations, the Prodigy/CME Machine-Controller card is only reset during power-up. There are, however, several other ways that the Prodigy/CME Machine-Controller card can be reset, many of which are not related to a power cycle. See Section 2.6.5, "Card Reset," for a description of how the card can be reset. Whether via a power up or some other event, a reset serves the purpose of initializing values and bringing the Prodigy/CME Machine-Controller card to a known and consistent state.

To determine the cause of a card reset, a C-Motion command can be sent to read the reset source. The following table details the encoding of this word, which can be read via the Reset Cause register of the Peripheral I/O space.

I/O Address	Bit Location	Signals
2	0-10	*Reserved*
	11	C-Motion Engine user application code fault. A 1 value in this bit indicates an instruction or address access fault.
	12	Commanded reset. A 1 value in this bit indicates a card-level reset commanded via a C-Motion command.
	13	Undervoltage detection: a 1 value in this bit indicates a reset caused by undervoltage detection.
	14	*Reserved*
	15	Magellan Watchdog timeout: a 1 value in this bit indicates a reset caused by the card watchdog timeout. See Section 2.3.4, "Card Watchdog Timer," for a description.

Note that after a card power cycle this register will always contain 0x2000, indicating an undervoltage condition.

The most common use of this feature is as a safety check, contained in user application code residing on the C-Motion Engine. Since user application code most often automatically executes after card powerup, a check of this register by the user application code will allow anomalous occurrences of a card reset to be flagged and investigated.

Another common use is to determine the nature of a Magellan reset. If an unexpected Magellan reset occurs, such as from a Magellan watchdog timeout, the user may use this register to assist with determining the cause.

2.6.4.1 C-Motion Commands

To read the machine controller's reset monitor register the command DeviceGetResetCause is used.

2.6.4.2 Connections & Associated Signals

There are no signals associated with this function.

2.6.5 Card Reset

Although a reset occurs automatically during power-up, it is sometimes desirable to reset the Prodigy/CME Machine -Controller card explicitly through a user-initiated command or action.

After a card reset occurs the Magellan Motion Processor and the C-Motion Engine modules will be reset, and many of the Prodigy/CME Machine-Controller card's output signals will be driven to known states. These are summarized in the following table:

Signal Name	State
AxisOut1-4	High
AnalogOut1-8	0.0 volts
DigitalOut1-4	Low
AmpEnable1-4	High

Digital I/O(1-4) default to be inputs at card reset.

In addition, upon a card reset all card default parameters are reloaded. See Section 2.6.6, "Setting Card Defaults," for more information on default values.

2.6.5.1 C-Motion Commands

The C-Motion command DeviceReset is used to reset the card.

2.6.5.2 Connections & Associated Signals

Although as noted above there are many card signals affected by a reset, there are no specific signals associated with this function.

2.6.6 Setting Card Defaults

There are a number of user-settable parameters that are saved by the card in non-volatile RAM and that are utilized after a powerup, or after a card reset. The following table shows these parameters, and provides the initial factory default values:

Parameter	Factory default value
Ethernet Communications	
IP Address	192.168.2.2
Net Mask	255.255.255.0
Gateway	0.0.0.0
PRP Port	40100
Serial Communications	
Serial 1 settings	57600, no parity, 8 data bits, 1 stop bit
Serial 2 settings	115200, no parity, 8 data bits, 1 stop bit

Parameter	Factory default value
RS485 duplex	Full
CANbus Communications	
Baud Rate	1,000,000
Send Address	0x580
Receive Address	0x600
Task Control & User Application Code	
Auto start (y/n)	No
Debug/Console channel	None

If desired, new default values can be stored by the user. Note that the updated defaults will only take effect after a card reset.

The default values are stored in the card's NVRAM area. As such, writing and reading operations to the default value area involves special considerations. See Section 2.6.2, "Non-volatile Memory," for more information on NVRAM memory operations.

2.6.6.1 C-Motion Commands

The C-Motion command DeviceSetDefault is used to set new default values. The command DeviceGetDefault is used to read these values back.

For detailed information on these C-Motion commands consult the *Prodigy/CME Programmer's Reference.*

2.6.7 Card Undervoltage Monitor

The Prodigy/CME Machine-Controller cards provides an internal logic undervoltage detection circuit. An undervoltage condition occurs when the card's 3.3V internal voltage drops below 2.7V.

If a card undervoltage condition occurs the card will be automatically reset. See Section 2.6.5, "Card Reset," for more information on the signal and other conditions affected by a card reset. See Section 2.6.4, "Reset Monitor," to determine if a reset was caused by an undervoltage condition.

An undervoltage fault is a very serious condition. On the occurrence of an undervoltage fault the user should immediately power down the card and determine the cause of the fault before repowering the card.

2.6.7.1 C-Motion Commands

There are no user-settable parameters associated with this feature. See Section 2.6.4, "Reset Monitor," for a description of the reset monitor, and how to determine what type of reset occurred.

2.6.8 Card Resource Information

There are various resources on the machine controller that contain programmable or otherwise changeable logic. It may be beneficial to the user to be aware of revision information associated with these logic or firmware entities.

In general, once the user has developed his motion application they may want to query and record the card resource revision information so that they can be aware of any changes to these revisions in future production units purchased from PMD, or to allow them to order specific revisions.

Along these lines, the Prodigy/CME Machine-Controller card resource information that may be queried is the card logic version, the device firmware version, and the dual-ported RAM size.

2.6.8.1 C-Motion Commands

To query the card logic version the C-motion command DeviceGetLogicVersion is used. To query the device firmware version the command DeviceGetVersion is used, and to query the DP-RAM size the command DeviceGetRAMSize is used.

2.7 C-Motion Engine Functions

The C-Motion Engine on the Prodigy/CME Machine-Controller card allows C-Motion code to be downloaded and executed on the card. The C-Motion Engine is a powerful and flexible engine that can be used to:

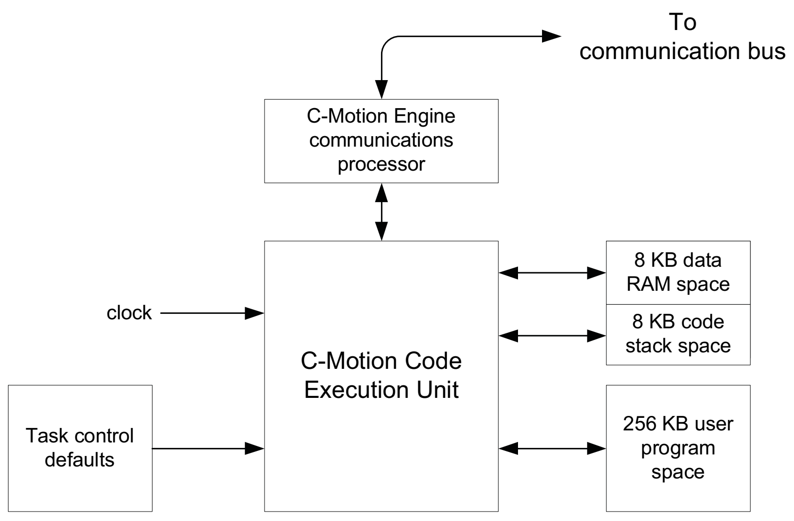
- Operate Prodigy motion cards in a standalone mode
- Offload time-critical code from the host to the motion card
- Create a complete machine controller that communicates via serial, CANbus, or Ethernet to a cell controller or other high level controller
- Extend the functionality of the Magellan Motion Processor with higher level functions such as contouring, macros, or other complex behaviors
- Lower system cost by combining a motherboard function with a dedicated motion card function in a single-card format.

2.7.1 C-Motion Engine Hardware Configuration

The C-Motion Engine is a self-contained module that provides non-volatile RAM space to store downloaded user application code, RAM space for 'scratch' data variable storage, and connections to the communication bus allowing the C-Motion Engine to send and receive messages through the network ports, communicate with the Magellan Motion Processor, and access other on-card resources such as the dual-ported RAM.

Creating, compiling, downloading, and verifying a specific user C-Motion application on a Prodigy/CME Machine-Controller card is accomplished with the C-Motion Engine development system, described in *C-Motion Engine Development Tools.* The outcome of such a development sequence is a downloadable code image, run on the C-Motion Engine, that contains the user application code and that is executed by the C-Motion Engine on the Prodigy/CME Machine-Controller card. Figure 2-20 provides an overview of the architecture of the C-Motion Engine.

Figure 2-20: Overview of C-Motion Engine Architecture


To communication bus
C-Motion Engine communications processor
clock
C-Motion Code Execution Unit
8 KB data RAM space
8 KB code stack space
Task control defaults
256 KB user program space

The following table provides an operational overview of the capabilities and resources provided by the C-Motion Engine:

Resource	Specification
MIPS (millions of instructions per second)	96
User program space (stored in flash)	256KB
User data RAM space	8 KB
User code stack space	8 KB

2.7.2 Powerup & Operation

Upon reset or powerup the C-Motion Engine initializes itself and checks to see whether execution of user application code, if downloaded, should automatically begin. If the factory default settings have not been changed, user application code automatically begins executing and continues until the card is powered down or until a specific ‘stop executing’ command is given. If this default value is changed, then the C-Motion Engine will hold in a wait state, and code execution will not occur automatically.

While there are numerous safety checks and features built into the C-Motion Engine system, application code developed for the C-Motion Engine is C-based, and thus there are limits to code size, RAM usage, and stack usage that should be observed during runtime operation of downloaded C-Motion code. The table above provides these numerical limits.

For user downloaded code that does not correctly observe these limits, or for files that have become corrupted, there are a number of fault conditions that can occur while the C-Motion Engine is executing downloaded user application code. These very serious run-time faults include instruction errors- indicating that an unknown instruction was encountered during execution of the user’s code, and address faults- indicating that either a program space or RAM space access limit was violated. If either of these conditions occur, the C-Motion Engine will immediately halt user code execution, and reset the Prodigy/CME Machine-Controller card. This C-Motion Engine-initiated reset is identical to the reset that occurs after sending a card reset C-Motion command, except that the cause of the reset is recorded as ‘C-Motion Engine user code Fault’ rather than ‘commanded’ reset. See Section 2.6.5, “Card Reset,” for more information on the Reset command. See Section 2.6.4, “Reset Monitor,” for information on retrieving the reset cause.

Whether or not user application code is running, after reset or powerup, the C-Motion Engine begins processing PRP actions sent to it. These commands are typically sent from a host controller. The supported commands include functions such as checking the downloaded user application code version stored in the C-Motion Engine, and sending and receiving messages to the user code loaded on to the C-Motion Engine.

For additional guidelines on managing run-time usage of the C-Motion Engine see the *C-Motion Engine Development Tools* manual.

2.7.3 Task Control

The primary purpose of the C-Motion Engine is to execute user application code that has been downloaded to it using the C-Motion Engine development system.

In a production environment, this code will typically automatically start upon powerup, and run continuously while the system is in operation. For debugging however, there are a number of additional controls.

At any point in time it is possible to stop or restart execution of the C-Motion Engine user application code. To access this function the C-Motion command actions TaskStart and TaskStop are used. To read the current status of the task the command GetState is used.

Extreme caution should be applied when stopping or starting user application code running on the C-Motion Engine, as depending on the specific application code, this may cause unexpected or unsafe motion. It is the responsibility of the user to determine whether stopping or restarting of user application code is safe and appropriate.

Whether or not the user application code automatically executes upon powerup or reset can also be controlled. The two options are operation under manual mode, in which case the user's code will not begin execution until an explicit start command is given, and auto-start, where the code automatically begins execution from powerup or reset.

This setting is part of the card default parameters, so to change these parameters the C-Motion command DeviceSetDefault is used. See Section 2.6.6, "Setting Card Defaults" for more information on default settings and how to program them.

For a detailed description of the supported Prodigy/CME commands see the *PMD Resource Access Protocol Programmer's Reference.*

2.7.4 Sending Messages to/from User Application Code

A common function of user application code running on the C-Motion Engine is to parse command messages sent to it by a host controller. For example a user might write code for the C-Motion Engine that responds to an "Extend RobotArm" command sent by the host controller, and then send a series of commands to the Magellan Motion Processor to execute this motion sequence. At the end of the motion sequence the user application code might send an "Arm Extended" message confirming the movement sequence has completed.

One method of achieving this is to use the Prodigy/CME Machine-Controller card's peripheral mechanism to open, and operate, a low-level communications link via the serial, CANbus, or Ethernet link. This method has the advantage of giving relatively direct control over the communication traffic. The disadvantage is that the user has to implement specific send and receive communications in the host controller, and the C-Motion Engine needs to have similar code implemented that can process these messages.

Another method that may be more convenient, particularly during early debugging of the user's application code, is to use a capability of the PRP system to connect directly to the user application code on the C-Motion Engine. Messages sent and received by the C-Motion Engine from a host controller are stored in a special buffer, and can be easily read or written to by the user application code. In addition, PMD's Pro-Motion application supports a simple way of entering, sending, and/or receiving such messages. This makes it easy to manually enter commands from Pro-Motion and exercise the user application code which is programmed to parse these messages.

To utilize this approach, a new peripheral is opened using the C-Motion command PeriphOpenCME. Thereafter PeriphSend or PeriphReceive commands are used to send and receive these messages within the C-Motion Engine.

In addition to these communications commands, the C-Motion command DeviceNoOperation performs a basic connection check. A return without an error code indicates that the C-Motion Engine is accessible and processing commands.

2.7.5 Connecting To The C-Motion Engine Code During Development

To develop code that is downloaded to the C-Motion Engine a communications link is required between the C-Motion Engine and the PC-based C-Motion Engine development environment. This link contains the information required for code downloads, as well as other information utilized during application debugging.

While serial, CANbus, or Ethernet can all be used as the communications link, typically, this link is chosen to be Ethernet because it is significantly faster than the other two. If the production machine network is also Ethernet, then only one network need be used for both code development, and operation of the machine.

It is also possible to use separate communication channels, so that one type of link is used for code development and download, and another for the 'application' communications. This has the advantage that application and code development traffic are not intermingled. For example, in a production machine control application that involves a PC and two Prodigy/CME Machine-Controller cards communicating via CANbus, Ethernet can be used as the development link, while the application software contained in the C-Motion code will send and receive messages using the CANbus port.

Most combinations of application and download/debug link selection are allowed. However the exception is the serial port. If one serial channel is used as the application link, the other serial port, or the CANbus or Ethernet port, must be used for the download link. This is because unlike CANbus and Ethernet ports, serial ports do not support multiple 'conversations' within a single physical port.

Selecting which Prodigy/CME Machine-Controller channel will be used for download is specified via the C-Motion development system. For more information see the *C-Motion Engine Development Tools* manual.

2.7.6 Debug Console Window

During development, the user can use procedure calls similar to **printf()** from the downloaded application on the C-Motion Engine to send messages to the PC Development Environment for display in a special console window. These console messages may be useful for checking code progress, displaying internal variables, or for other code development-related purposes.

The default console channel is none, however this can be changed using the DeviceSet command with the Device resource specified as CANbus or Ethernet/UDP.

2.7.7 Downloading and Verifying User Application Code

The C-Motion Engine development system is used to create, compile, and download user application code. The development system can download the file image for the current code project being worked on, or a specific named file can be downloaded. Downloaded files images end with a ".bin" extension. Only one code image file may be downloaded into the C-Motion Engine at a time. Downloading a new image automatically erases the previous code image.

There are times when it may be useful to read specific characteristics of a code file that has been downloaded into the C-Motion Engine. For example a host controller in a production environment may want to confirm that the host application code version actually loaded on the C-Motion Engine matches the expected production code version. To accomplish this, the C-Motion command DeviceGet is used.

Using this command the file name of the downloaded user application code, the checksum of the downloaded file, the date & time of file creation, and the version number of the C-Motion Engine itself (loaded by PMD at the Prodigy/CME factory) can be retrieved.

For complete information on the format and function of these, and other C-Motion calls refer to the *PMD Resource Access Protocol Programmer's Reference.*

2.7.8 C-Motion Engine Heartbeat LED

The Prodigy/CME Machine-Controller card utilizes an LED, labeled D3, locatable using Figure 1-5 to provide visual confirmation of C-Motion Engine activity. Two different states can be distinguished, user application code running, and user application code not running.

User application code running means that a file has been downloaded and is actively being executed by the C-Motion Engine. This is indicated by a steady on/off blinking of the LED, once per second.

If no user application code has been downloaded, or if code execution has been halted by the user or for some other reason, the LED changes to a 'chirp' indication, with a blinking pattern consisting of very brief on, followed by one second off.

2.8 Software Libraries

PMD provides software libraries in C/C++ as well as Visual Basic to access all of the functions provided by the Prodigy/ CME Machine-Controller cards. In addition, a special library of commands is used to access Magellan Motion Processor functions called VB-Motion, and C-Motion.

The *PMD Resource Access Protocol Programmer's Reference*, in addition to providing detailed format and function descriptions of the PRP messages associated with the Prodigy/CME Machine-Controller cards, also provides the equivalent C and Visual Basic library calls. There is often, but not always, a direct correspondence between PRP messages and C language software commands as they will be used for user application code to be executed in the host controller, or in the C-Motion Engine.

2.8.1 C Language Code Running on the C-Motion Engine

One of the most powerful features of the software libraries provided for the Prodigy/CME Machine-Controller card is that the code sequence used to accomplish a specific function such as accessing the dual-ported RAM, or sending commands to the Magellan Motion Processor, does not change whether the code is compiled for execution on the host controller, or for execution on the C-Motion Engine.

This allows motion control applications developed using C-Motion on a host computer to be easily ported to the Prodigy/CME Machine-Controller card. The C-Motion Engine provides a C programming environment for motor control without requiring the time-consuming development of the entire embedded framework. Only the part unique to a specific motor control application must be provided.

3. Electrical Reference

In This Chapter

3.1 User-Settable Components

Figure 3-1 illustrates the locations of the principal components of the Prodigy/CME Machine-Controller cards. The user-settable components of the card are listed in the following table:

Component	Function
Resistor packs JS1, JS2, and JS3	Sets the encoder termination.

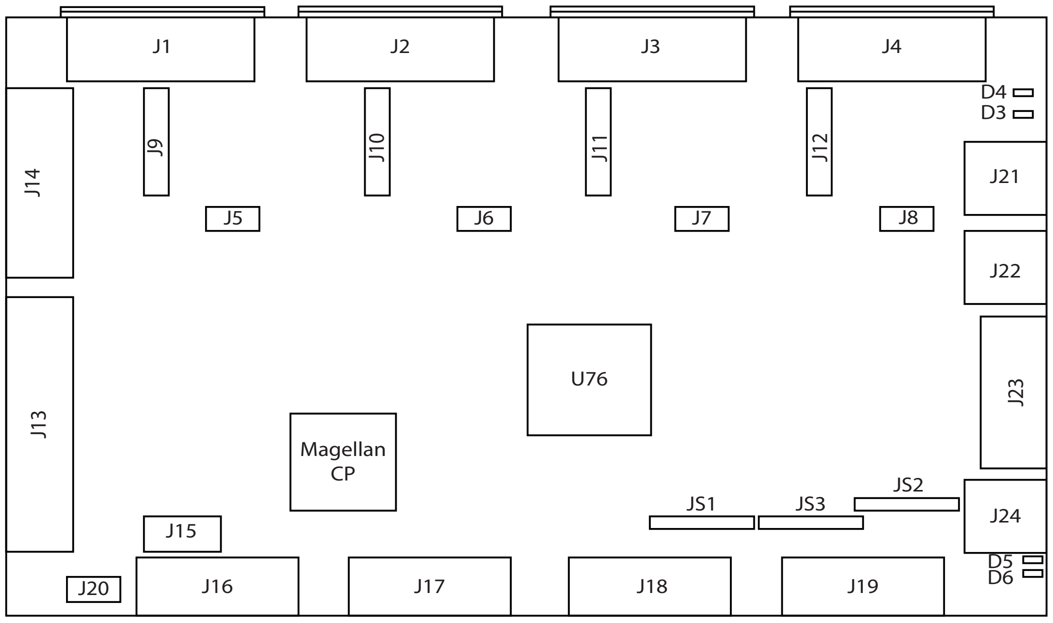


Figure 3-1: Components and Layout, Front of Card

3.1.1 Encoder Connections and Resistor Packs

Encoder inputs may be connected differentially, with two wires for *QuadA*, *QuadB*, and *Index* signals, or with just one wire per signal. If differential connections are being used, resistor packs JS1, JS2, and JS3 should remain installed. If single-ended encoders are used, remove all three resistor packs, and connect encoder signals to the positive encoder input only. The negative input may remain unconnected.

The following table shows the relationship between the encoder input mode and resistor packs:

Item	Setting	Description
Resistor packs JS1, JS2, JS3	Installed; this is the default setting of resistor packs JS1 - JS3.	If differential connections are being used, leave the resistor packs installed.
	Removed	If single-ended encoder connections are being used, remove the resistor packs.

Encoder connections when using differential encoder input:

Signal	J16 – J19			
	Axis 1	Axis 2	Axis 3	Axis 4
QuadA*n*+	J16-1	J17-1	J18-1	J19-1
QuadA*n*-	J16-2	J17-2	J18-2	J19-2
QuadB*n*+	J16-3	J17-3	J18-3	J19-3
QuadB*n*-	J16-4	J17-4	J18-4	J19-4
Index*n*+	J16-6	J17-6	J18-6	J19-6
Index*n*-	J16-7	J17-7	J18-7	J19-7
Vcc	J16-14	J17-14	J18-14	J19-14
GND	J16-15	J17-15	J18-15	J19-15

Encoder connections when using single-ended encoder input:

Signal	J16 – J19			
	Axis 1	Axis 2	Axis 3	Axis 4
QuadA*n*	J16-1	J17-1	J18-1	J19-1
QuadB*n*	J16-3	J17-3	J18-3	J19-3
Index*n*	J16-6	J17-6	J18-6	J19-6
Vcc	J16-14	J17-14	J18-14	J19-14
GND	J16-15	J17-15	J18-15	J19-15

3.2 Connectors

There are 24 user-accessible connectors on the Prodigy/CME Machine-Controller card. See Figure 3-1 for the specific locations of the connectors on the card. The connectors and their functions are outlined in the following table:

Connector Name	Connector #	Functionality
Axis 1 Atlas	J1	Provides socket for Axis 1 pluggable Atlas unit.
Axis 2 Atlas	J2	Provides socket for Axis 2 pluggable Atlas unit.
Axis 3 Atlas	J3	Provides socket for Axis 3 pluggable Atlas unit.
Axis 4 Atlas	J4	Provides socket for Axis 4 pluggable Atlas unit.
Axis 1 Power	J5	Provides power to the whole card plus Axis 1 Atlas unit. It accepts DC supply in the range of +12 – 56 VDC.

Connector Name	Connector #	Functionality
Axis 2 Power	J6	Provides power to the Axis 2 Atlas unit. It accepts DC supply in the range of +12 – 56 VDC.
Axis 3 Power	J7	Provides power to the Axis 3 Atlas unit. It accepts DC supply in the range of +12 – 56 VDC.
Axis 4 Power	J8	Provides power to the Axis 4 Atlas unit. It accepts DC supply in the range of +12 – 56 VDC.
Axis 1 Motor Drive	J9	Provides direct motor control signals for Axis 1.
Axis 2 Motor Drive	J10	Provides direct motor control signals for Axis 2.
Axis 3 Motor Drive	J11	Provides direct motor control signals for Axis 3.
Axis 4 Motor Drive	J12	Provides direct motor control signals for Axis 4.
General I/O	J13	Provides the following general I/O interfaces to the board: 8 channels of bi-directional digital I/O; 4 channels of digital inputs; 4 channels of digital outputs; 8 channels of differential analog.
Amplifier I/O	J14	Provides amplifier enable outputs for 4 axes. Provides 8 AnalogOutput signals for controlling external amplifiers or general purpose ±10V analog output. Provides AxisIn and AxisOut connections for all four axes.
Expansion	J15	Provides synchronization of multiple Prodigy Machine-Controller boards within a single system.
Axis 1 Feedback	J16	Provides differential or single-ended motor encoder feedback signals such as Quad A/B, Index, Hall A/B/C, PosLim and NegLim for Axis1.
Axis 2 Feedback	J17	Provides differential or single-ended motor encoder feedback signals such as Quad A/B, Index, Hall A/B/C, PosLim and NegLim for Axis 2.
Axis 3 Feedback	J18	Provides differential or single-ended motor encoder feedback signals such as Quad A/B, Index, Hall A/B/C, PosLim and NegLim for Axis 3.
Axis 4 Feedback	J19	Provides differential or single-ended motor encoder feedback signals such as Quad A/B, Index, Hall A/B/C, PosLim and NegLim for Axis 4.
+5V Power	J20	Provides logic power to the board when HV1 is not available.
CAN1, CAN2	J21, J22	RJ45 connector that provide connection to a CAN 2.0B network.
Serial	J23	DB-9 serial port for RS232 or RS485 connections.
Ethernet	J24	RJ45 connector that provide connection to an Ethernet TCP/IP network

3.2.1 Atlas Connector

The Atlas Connector (J1 – J4) is the socket that accepts Atlas, the vertical single-axis digital amplifier unit.

3.2.1.1 Axis 1 Atlas Socket

Pin	Connection	Description
		J1
1	GND	Power return for HV1, Motor A, Motor B, Motor C and Motor D.
2	GND	Power return for HV1, Motor A, Motor B, Motor C and Motor D.
3	HV1	DC power to axis 1 Atlas module, referenced to GND.
4	HV1	DC power to axis 1 Atlas module, referenced to GND.
5	Motor A1	Axis 1 motor output signal A.
6	Motor A1	Axis 1 motor output signal A.
7	Motor B1	Axis 1 motor output signal B.
8	Motor B1	Axis 1 motor output signal B.
9	Motor C1	Axis 1 motor output signal C.
10	Motor C1	Axis 1 motor output signal C.
11	Motor D1	Axis 1 motor output signal D.
12	Motor D1	Axis 1 motor output signal D.
13	~Enable1	An active-low enable signal.
14	FaultOut1	FaultOut1 provides programmable fault indication.
15	NC	Not Connected.
16	GND	Ground return for ~Enable, FaultOut, SPI or pulse & direction signals.
17	~SPICS1	Atlas SPI bus slave select1.
18	SPISI	Atlas SPI bus master output, slave input.
19	SPICLK	Atlas SPI bus serial clock.
20	SPISO	Atlas SPI bus master input, slave output.

3.2.1.2 Axis 2 Atlas Socket

Pin	Connection	Description
		J2
1	GND	Power return for HV2, Motor A, Motor B, Motor C and Motor D.
2	GND	Power return for HV2, Motor A, Motor B, Motor C and Motor D.
3	HV2	DC power to Axis 2 Atlas module, referenced to GND.
4	HV2	DC power to Axis 2 Atlas module, referenced to GND.
5	Motor A2	Axis 2 motor output signal A.
6	Motor A2	Axis 2 motor output signal A.
7	Motor B2	Axis 2 motor output signal B.
8	Motor B2	Axis 2 motor output signal B.
9	Motor C2	Axis 2 motor output signal C.
10	Motor C2	Axis 2 motor output signal C.
11	Motor D2	Axis 2 motor output signal D.
12	Motor D2	Axis 2 motor output signal D.
13	~Enable2	An active-low enable signal.
14	FaultOut2	FaultOut2 provides programmable fault indication.
15	NC	Not Connected.
16	GND	Ground return for ~Enable, FaultOut, SPI or pulse & direction signals.
17	~SPICS2	Atlas SPI bus slave select2.
18	SPISI	Atlas SPI bus master output, slave input.
19	SPICLK	Atlas SPI bus serial clock.
20	SPISO	Atlas SPI bus master input, slave output.

3.2.1.3 Axis 3 Atlas Socket

Pin	Connection	Description
		J3
1	GND	Power return for HV3, Motor A, Motor B, Motor C and Motor D.
2	GND	Power return for HV3, Motor A, Motor B, Motor C and Motor D.
3	HV3	DC power to the Axis 3 Atlas module, referenced to GND.
4	HV3	DC power to the Axis 3 Atlas module, referenced to GND.
5	Motor A3	Axis 3 motor output signal A.
6	Motor A3	Axis 3 motor output signal A.
7	Motor B3	Axis 3 motor output signal B.
8	Motor B3	Axis 3 motor output signal B.
9	Motor C3	Axis 3 motor output signal C.
10	Motor C3	Axis 3 motor output signal C.
11	Motor D3	Axis 3 motor output signal D.
12	Motor D3	Axis 3 motor output signal D.
13	~Enable3	An active-low enable signal.
14	FaultOut3	FaultOut3 provides programmable fault indication.
15	NC	Not Connected.
16	GND	Ground return for ~Enable, FaultOut, SPI or pulse & direction signals.
17	~SPICS3	Atlas SPI bus slave select3.
18	SPISI	Atlas SPI bus master output, slave input.
19	SPICLK	Atlas SPI bus serial clock.
20	SPISO	Atlas SPI bus master input, slave output.

3.2.1.4 Axis 4 Atlas Socket

Pin	Connection	Description
		J4
1	GND	Power return for HV4, Motor A, Motor B, Motor C and Motor D.
2	GND	Power return for HV4, Motor A, Motor B, Motor C and Motor D.
3	HV4	DC power to Axis 4 Atlas module, referenced to GND.
4	HV4	DC power to Axis 4 Atlas module, referenced to GND.
5	Motor A4	Axis 4 motor output signal A.
6	Motor A4	Axis 4 motor output signal A.
7	Motor B4	Axis 4 motor output signal B.
8	Motor B4	Axis 4 motor output signal B.
9	Motor C4	Axis 4 motor output signal C.
10	Motor C4	Axis 4 motor output signal C.
11	Motor D4	Axis 4 motor output signal D.
12	Motor D4	Axis 4 motor output signal D.
13	~Enable4	An active-low enable signal.
14	FaultOut4	FaultOut4 provides programmable fault indication.
15	NC	Not Connected.
16	GND	Ground return for ~Enable, FaultOut, SPI or pulse & direction signals.
17	~SPICS4	Atlas SPI bus slave select4.
18	SPISI	Atlas SPI bus master output, slave input.
19	SPICLK	Atlas SPI bus serial clock.
20	SPISO	Atlas SPI bus master input, slave output.

3.2.2 Power Connector

For each Atlas unit installed on the Prodigy/CME Machine-Controller card, there will be a dedicated 2 pin HV power connector that accepts input voltage in the range of +12 – 56VDC. J5, J6, J7 and J8 are used to provide power to each Atlas unit. Molex 4.20mm Pitch Mini-Fit plus HCS series, single row, vertical, Power Terminal Headers are used for the 2-pin power connectors.

Note that there is a +5V power connector (J20) on the board. Users can power the board with +5V supply when HV1 is not available. However, this will not power the Atlas modules. When HV1 is present, the +5V supply via J20 should not be used. J20 uses the Molex 3.00 Pitch Micro-Fit Header, 2-signal, single row, vertical connector.

3.2.2.1 Axis 1 Power Connector

Pin	Connection	Description
		J5
1	HV1	Provides DC power to the card and Axis 1 Atlas module
2	GND	Ground

3.2.2.2 Axis 2 Power Connector

Pin	Connection	Description
		J6
1	HV2	Provides DC power to the Axis 2 Atlas module
2	GND	Ground

3.2.2.3 Axis 3 Power Connector

Pin	Connection	Description
		J7
1	HV3	Provides DC power to the Axis 3 Atlas module
2	GND	Ground

3.2.2.4 Axis 4 Power Connector

Pin	Connection	Description
		J8
1	HV4	Provides DC power to the Axis 4 Atlas module
2	GND	Ground

3.2.3 Feedback Connector

The Feedback Connector (J16 – J19 in Figure 3-1) provides connections to various motor feedback signals. The Feedback Connector uses the 15-pin high density DB connector, which can be connected to PMD Cable-5005-01. Please refer to Section 3.4, "Cables" for detailed information on Prodigy/CME Machine-Controller cable specifications.

For brushless DC motors, it also connects the Hall effect signals typically used to commutate the motor. The Halls are not used with the DC brush or step motors.

3.2.3.1 Axis 1 Feedback Connector

Pin	Connection	Description
		J16
1	QuadA1+	Axis 1 Quadrature A+ encoder input
2	QuadA1-	Axis 1 Quadrature A- encoder input
3	QuadB1+	Axis 1 Quadrature B+ encoder input
4	QuadB1-	Axis 1 Quadrature B - encoder input
5	GND	Ground
6	Index1+	Axis 1 Index+ input
7	Index1-	Axis 1 Index1- input
8	HallA1	Axis 1 Hall A input
9	HallB1	Axis 1 Hall B input
10	HallC1	Axis 1 Hall C input
11	Home1	Axis 1 Home input
12	PosLim1	Axis 1 Positive direction limit switch input
13	NegLim1	Axis 1 Negative direction limit switch input
14	Vcc	+5V
15	GND	Ground

3.2.3.2 Axis 2 Feedback Connector

Pin	Connection	Description
		J17
1	QuadA2+	Axis 2 Quadrature A+ encoder input
2	QuadA2-	Axis 2 Quadrature A- encoder input
3	QuadB2+	Axis 2 Quadrature B+ encoder input
4	QuadB2-	Axis 2 Quadrature B- encoder input
5	GND	Ground
6	Index2+	Axis 2 Index+ input
7	Index2-	Axis 2 Index- input
8	HallA2	Axis 2 Hall A input
9	HallB2	Axis 2 Hall B input
10	HallC2	Axis 2 Hall C input
11	Home2	Axis 2 Home input
12	PosLim2	Axis 2 Positive direction limit switch input
13	NegLim2	Axis 2 Negative direction limit switch input
14	Vcc	+5V
15	GND	Ground

3.2.3.3 Axis 3 Feedback Connector

Pin	Connection	Description
		J18
1	QuadA3+	Axis 3 Quadrature A+ encoder input
2	QuadA3-	Axis 3 Quadrature A- encoder input
3	QuadB3+	Axis 3 Quadrature B+ encoder input
4	QuadB3-	Axis 3 Quadrature B- encoder input
5	GND	Ground
6	Index3+	Axis 3 Index+ input
7	Index3-	Axis 3 Index- input

8	HallA3	Axis 3 Hall A input
9	HallB3	Axis 3 Hall B input
10	HallC3	Axis 3 Hall C input
11	Home3	Axis 3 Home input
12	PosLim3	Axis 3 Positive direction limit switch input
13	NegLim1	Axis 3 Negative direction limit switch input
14	Vcc	+5V
15	GND	Ground

3.2.3.4 Axis 4 Feedback Connector

Pin	Connection	Description
		J19
1	QuadA4+	Axis 4 Quadrature A+ encoder input
2	QuadA4-	Axis 4 Quadrature A- encoder input
3	QuadB4+	Axis 4 Quadrature B+ encoder input
4	QuadB4-	Axis 4 Quadrature B- encoder input
5	GND	Ground
6	Index4+	Axis 4 Index+ input
7	Index4-	Axis 4 Index- input
8	HallA4	Axis 4 Hall A input
9	HallB4	Axis 4 Hall B input
10	HallC4	Axis 4 Hall C input
11	Home4	Axis 4 Home input
12	PosLim4	Axis 4 Positive direction limit switch input
13	NegLim4	Axis 4 Negative direction limit switch input
14	Vcc	+5V
15	GND	Ground

3.2.4 Motor Drive Connector

The Motor Drive connectors (J9 – J12) provide motor output signals for use with brushless DC, DC brush, or step motors. These are Molex 4.20mm Pitch Mini-Fit Plus HCS series, single row, vertical connectors.

3.2.4.1 Axis 1 Motor Connector

Pin	Connection	Description
		J9
1	Motor A1	Axis 1 motor output signal A.
2	Motor B1	Axis 1 motor output signal B.
3	Motor C1	Axis 1 motor output signal C.
4	Motor D1	Axis 1 motor output signal D.
5	GND	Ground

3.2.4.2 Axis 2 Motor Connector

Pin	Connection	Description
		J10
1	Motor A2	Axis 2 motor output signal A.
2	Motor B2	Axis 2 motor output signal B.
3	Motor C2	Axis 2 motor output signal C.
4	Motor D2	Axis 2 motor output signal D.
5	GND	Ground

3.2.4.3 Axis 3 Motor Connector

Pin	Connection	Description
		J11
1	Motor A3	Axis 3 motor output signal A.
2	Motor B3	Axis 3 motor output signal B.
3	Motor C3	Axis 3 motor output signal C.
4	Motor D3	Axis 3 motor output signal D.
5	GND	Ground

3.2.4.4 Axis 4 Motor Connector

Pin	Connection	Description
		J12
1	Motor A4	Axis 4 motor output signal A.
2	Motor B4	Axis 4 motor output signal B.
3	Motor C4	Axis 4 motor output signal C.
4	Motor D4	Axis 4 motor output signal D.
5	GND	Ground

3.2.5 General I/O Connector

The General I/O connector provides various I/O connections to the board. There are 8 channels of bi-directional digital I/O; 4 channels of digital input; 4 channels of digital output and 8 channels of differential analog inputs. This is a high density DB-44 connector.

Pin	Connection	Description	Pin	Connection	Description
			J13		
1	AnalogIn1+	Analog input 1+	2	AnalogIn1-	Analog input 1-
3	AnalogIn2+	Analog input 2+	4	AnalogIn2-	Analog input 2-
5	AGND	Ground for analog signals	6	AGND	Ground for analog signals
7	Reserved	Reserved signal	8	Reserved	Reserved signal
9	DigitalIO1	Digital input/output 1	10	DigitalIO2	Digital input/output 2
11	DigitalIO3	Digital input/output 3	12	GND	Ground
13	DigitalOut1	Digital output 1	14	DigitalOut2	Digital output 2
15	DigitalOut3	Digital output 3	16	AnalogIn3+	Analog input 3+
17	AnalogIn4+	Analog input 4+	18	AnalogIn5+	Analog input 5+
19	AnalogIn6+	Analog input 6+	20	AnalogIn7+	Analog input 7+
21	AnalogIn8+	Analog input 8+	22	AGND	Ground for analog signals
23	Reserved	Reserved signal	24	Reserved	Reserved signal
25	DigitalIO4	Digital input/output 4	26	DigitalIO5	Digital input/output 5

27	DigitalIO6	Digital input/output 6	28	DigitalIn1	Digital input 1
29	DigitalIn2	Digital input 2	30	DigitalOut4	Digital output 4
31	AnalogIn3-	Analog input 3-	32	AnalogIn4-	Analog input 4-
33	AnalogIn5-	Analog input 5-	34	AnalogIn6-	Analog input 6-
35	AnalogIn7-	Analog input 7-	36	AnalogIn8-	Analog input 8-
37	AGND	Ground for analog signals	38	GND	Ground
39	GND	Ground	40	DigitalIO7	Digital input/output 7
41	DigitalIO8	Digital input/output 8	42	GND	Ground
43	DigitalIn3	Digital input 3	44	DigitalIn4	Digital input 4

3.2.6 Amplifier I/O Connector

The Amplifier I/O connector provides various amplifier-related signals including amplifier enable outputs, 8 AnalogOutput signals, AxisIn, and AxisOut signals. This is a high density DB-26 connector.

Pin	Connection	Description	Pin	Connection	Description
			J14		
1	AmpEnable1	Axis 1 amplifier enable signal	2	AmpEnable2	Axis 2 amplifier enable signal
3	GND	Ground	4	AxisIn1	Axis 1 AxisIn input
5	AxisIn2	Axis 2 AxisIn input	6	AGND	Ground for analog signals
7	AnalogOut1	Analog output 1	8	AnalogOut2	Analog output 2
9	AnalogOut3	Analog output 3	10	AmpEnable3	Axis 3 amplifier enable signal
11	AmpEnable4	Axis 4 amplifier enable signal	12	GND	Ground
13	AxisIn3	Axis 3 AxisIn input	14	AxisIn4	Axis 4 AxisIn input
15	AnalogOut4	Analog output 4	16	AnalogOut5	Analog output 5
17	AnalogOut6	Analog output 6	18	AGND	Ground for analog signals
19	AxisOut1	Axis 1 AxisOut output	20	AxisOut2	Axis 2 AxisOut output
21	AxisOut3	Axis 3 AxisOut output	22	AxisOut4	Axis 4 AxisOut output
23	GND	Ground	24	AGND	Ground for analog signals
25	AnalogOut7	Analog output 7	26	AnalogOut8	Analog output 8

3.2.7 Expansion Connector

This connector supports multi-card synchronization and an external RESET signal. Depending on whether the card is a master or slave, the SyncOut only or both SyncIn and SyncOut signals are used. This is a Molex 2.00 mini-pitch header, vertical, shrouded connector.

Pin	Connection	Description	Pin	Connection	Description
			J15		
1	Reserved	Reserved signal	2	Reserved	Reserved signal
3	Reserved	Reserved signal	4	SyncIn	Multiple cards SyncIn signal
5	GND	Ground	6	+5V	Maximum 100mA +5V output
7	Reserved	Reserved signal	8	Reserved	Reserved signal
9	Reserved	Reserved signal	10	SyncOut	Multiple cards SyncOut signal
11	GND	Ground	12	Reset	Active Low hardware reset

3.2.8 Serial Connector

The Serial Connector (J23) provides connections to two serial ports in RS232 mode, or a single serial port in RS485 mode. Electrically these connectors provide access to the same signals; however they have different physical connectors and wiring. The following sections provide information for the serial connector, and provide pinouts when operated in RS232 mode, RS485 full-duplex mode, and RS485 half-duplex mode.

Pin	Connection	RS232	RS485 Full Duplex	RS485 Half Duplex
			J23	
1	RS485Select	Open (default) selects RS232 mode for both Serial 1 and Serial2	When tied to ground selects RS485 mode for both Serial 1 and Serial2	When tied to ground selects RS485 mode for both Serial 1 and Serial2
2	Srl1Xmt/ RS485Xmt-	Serial 1 transmit output	Negative (inverting) transmit output	Negative (inverting) transmit/receive
3	Srl2Rcv/ RS485Rcv+	Serial 2 receive input	Positive (non-inverting) receive input	No Connect
4	No connect	No connect	No Connect	No Connect
5	GND	Ground	Ground	Ground
6	Srl2Rcv/ RS485Rcv+	Serial 2 receive input	Positive (non-inverting) receive input	No Connect
7	Srl1Rcv/ RS485Rcv-	Serial 1 receive input	Negative (inverting) receive input	No Connect
8	Srl2Xmt/ RS485Xmt+	Serial 2 transmit output	Positive (non-inverting) transmit output	Positive (non-inverting) transmit/receive
9	Srl1Xmt/ Rs485Xmt-	Serial 1 transmit output	Negative (inverting) transmit output	No connect
10	GND	Ground	Ground	Ground

3.2.9 CAN Connectors

The Prodigy/CME Machine-Controller's controller area network (CAN) transceivers are designed for use in applications employing the CAN serial communication physical layer in accordance with the ISO 11898 standard. The transceiver provides differential transmit and differential receive capability to/from a CAN controller at speeds up to 1 Mbps.

There are two different CAN connectors, J21, and J22, providing electrically identical signals. These two connectors are designed to make it easy to connect the Prodigy/CME Machine-Controller card in a daisy-chain configuration. Termination at each end of the cable run is generally recommended unless cable lengths are very short and speed is slow. ISO-11898 requires 120 Ohm termination at each end of the bus. Note that it is up to the customer to verify their network topology and operating parameters. The CANbus connector is a female RJ45 type connector.

See Section 2.4.2, "CANbus Communications" for more information on the functionality of the CANbus port.

The pinouts for both the J21 and J22 CAN connector are as follows:

Pin Number	Signal	Description
		J21, J22
1	CAN+	Positive CAN signal connection
2	CAN-	Negative CAN signal connection
3	GND	Ground
4	No Connect	Pass-through signal
5	No Connect	Pass-through signal
6	No Connect	Pass-through signal
7	GND	Ground
8	No Connect	Pass-through signal

3.2.10 Ethernet Connector

The Prodigy/CME Machine-Controller's Ethernet transceivers are designed for use with 10/100 base-TX Ethernet and support both TCP and UDP protocols. See Section 2.4.3, "Ethernet Communications" for more information on the functionality of the Ethernet port.

The Ethernet connector is an 8-pin female RJ45, and has two status LEDs, green and amber, which provide information on the status of the Ethernet link. A solid green LED indicates that a link exists. There is a transceiver connected on the 'other side' of the connection, and a blinking green LED means that data is being transmitted. The Amber LED indicates that a 100 Mbps (mega bits per second) network is in use. Without Amber LED indicates that the network is at 10 Mbps.

The pinouts for the J24 Ethernet connector are as follows:

Pin Number	Signal	Description
		J24
1	EthernetTx+	Ethernet differential transmit positive
2	EthernetTx-	Ethernet differential transmit negative
3	EthernetRx+	Ethernet differential receive positive
4	No connect	No connect
5	No connect	No connect
6	EthernetRx-	Ethernet differential receive negative
7	No connect	No connect
8	No connect	No connect

3.2.11 Connector Parts Reference

The following table is supplied as a reference only.

Label	Description	Connector Part Number	Connector Mate
J1-J4	Atlas Connector	Samtec SSQ-110-01-F-D	Pluggable Atlas Unit
J5-J8	Power Connector	Molex 46015-0203	Molex 39-01-2025
J23	Serial Connector	TE Connectivity -5747150-4	DB-9 Male
J9-J12	Motor Drive Connector	Molex 39-30-2050	Molex 39-01-4051
J22, J28	CAN Connector	(H) Amphenol FRJAE-408 (V) EDAC A00-108-620-450	Male RJ45
J24	Ethernet Connector	(H) Amphenol FRJAE-408 (V) EDAC A00-108-620-450	Male RJ45
J15	Expansion Connector	Molex 87831-1220	Molex 79107-7005
J16-19	Feedback Connector	FCI 10090929-S154VLF	High density DB-15 Male
J13	General I/O Connector	FCI 10090929-S444VLF	High density DB-44 Male
J14	Amplifier I/O Connector	FCI 10090929-S264VLF	High density DB-26 Male

3.3 Motor Connections

3.3.1 Atlas Mode

The following sections show the connections to different types of motors when the Atlas amplifiers are used.

3.3.1.1 Brushless DC Motor Connections

Connection Name	Axis 1	Axis 2	Axis 3	Axis 4
Motor A	J1-1	J2-1	J3-1	J4-1
Motor B	J1-2	J2-2	J3-2	J4-2
Motor C	J1-3	J2-3	J3-3	J4-3
Case/Shield	J1-5	J2-5	J3-5	J4-5

3.3.1.2 DC Brush Motor Connections

Connection Name	Axis 1	Axis 2	Axis 3	Axis 4
Motor+	J1-1	J2-1	J3-1	J4-1
Motor-	J1-3	J2-3	J3-3	J4-3
Case/Shield	J1-5	J2-5	J3-5	J4-5

3.3.1.3 Step Motor Connections

Connection Name	Axis 1	Axis 2	Axis 3	Axis 4
MotorA+	J1-1	J2-1	J3-1	J4-1
MotorA-	J1-2	J2-2	J3-2	J4-2
MotorB+	J1-3	J2-3	J3-3	J4-3
MotorB-	J1-4	J2-4	J3-4	J4-4
Case/Shield	J1-5	J2-5	J3-5	J4-5

3.3.2 Analog Output Mode

The following sections show the connection to different types of motors when the Analog output signals of the Amplifier I/O (J14) connector are used.

3.3.2.1 Brushless DC Motor Connections

Connection Name	Axis 1	Axis 2	Axis 3	Axis 4
AnalogOut1-4	J14-7	J14-8	J14-9	J14-15
AnalogOut5-8	J14-16	J14-17	J14-25	J14-26
AGND	J14-6	J14-6	J14-18	J14-24

3.3.2.2 DC Brush Motor Connections

Connection Name	Axis 1	Axis 2	Axis 3	Axis 4
AnalogOut1-4	J14-7	J14-8	J14-9	J14-15
AGND	J14-6	J14-6	J14-18	J14-24

3.3.2.3 Step Motor Connections

Connection Name	Axis 1	Axis 2	Axis 3	Axis 4
AnalogOut1-4	J14-7	J14-8	J14-9	J14-15
AnalogOut5-8	J14-16	J14-17	J14-25	J14-26
AGND	J14-6	J14-6	J14-18	J14-24

3.4 Cables

The following table provides a summary of the cables that are available with the Prodigy/CME Machine-Controller.

Component Part Number	Description
Cable-5001-01	2-signal HV Power supply cable. This stub cable provides power to the Atlas unit for each axis. This cable may plug into the Prodigy/CME Machine-Controller card's J5 – J8 connectors.
Cable-5002-01	5-signal Motor Drive cable. This stub cable connects to the Motor Drive Connectors.
Cable-5003-01	44-signal General I/O cable. This cable provides connections for the high density DB-44 connector (J13).
Cable-5004-01	26-signal Amplifier I/O cable. This cable provides connections for the high density DB-26 connector (J14).
Cable-4705-KIT-01.R	CANbus connector and terminator. This cable connects to the card's CANbus connector and has RJ45 connectors on both ends.
Cable-RJ45-02-R	Ethernet connector. This cable connects to the card's Ethernet connector.
Cable-1007-01	Serial connector. This cable connects to the card's serial connector.

The following sections provide detailed information on each of these cables.

3.4.1 Cable-5001-01

PMD Part #: Cable-5001-01

Description: 2-signal HV Power supply stub cable.

Length: 1.0 meter

J5-J8	Signal
1	HV
2	Ground

3.4.2 Cable-5002-01

PMD Part #: Cable-5002-01

Description: This stub cable connects to the Motor Drive Connectors.

Length: 1.0 meter

Cable: 5 conductor, 4.20mm pitch, 16AWG stranded cables

J9-J12 Pin	Signal	Wire Color
1	Motor A+	Red
2	Motor A-	White
3	Motor B+	Black
4	Motor B-	Brown
5	GND	Green

3.4.3 Cable-5003-01

PMD Part #: Cable-5003-01
Description: Male to female straight-through high density DB-44 connector cable.
Length: 1.0 meter
Cable: 44 conductor with 26AWG stranded cables.

J13 Pin	Signal	J13 Pin	Signal
1	AnalogIn1+	2	AnalogIn1-
3	AnalogIn2+	4	AnalogIn2-
5	AGND	6	AGND
7	Reserved	8	Reserved
9	DigitalIO1	10	DigitalIO2
11	DigitalIO3	12	GND
13	DigitalOut1	14	DigitalOut2
15	DigitalOut3	16	AnalogIn3+
17	AnalogIn4+	18	AnalogIn5+
19	AnalogIn6+	20	AnalogIn7+
21	AnalogIn8+	22	AGND
23	Reserved	24	Reserved
25	DigitalIO4	26	DigitalIO5
27	DigitalIO6	28	DigitalIn1
29	DigitalIn2	30	DigitalOut4
31	AnalogIn3-	32	AnalogIn4-
33	AnalogIn5-	34	AnalogIn6-
35	AnalogIn7-	36	AnalogIn8-
37	AGND	38	GND
39	GND	40	DigitalIO7
41	DigitalIO8	42	GND
43	DigitalIn3	44	DigitalIn4

3.4.4 Cable-5004-01

PMD Part #: Cable-5004-01

Description: Male to female straight through high density DB-26 connector cable.

Length: 1.0 meter

Cable: 26 conductor with 26AWG stranded cables.

J14 Pin	Signal	J14 Pin	Signal
1	AmpEnable1	2	AmpEnable2
3	GND	4	AxisIn1
5	AxisIn2	6	AGND
7	AnalogOut1	8	AnalogOut2
9	AnalogOut3	10	AmpEnable3
11	AmpEnable4	12	GND
13	AxisIn3	14	AxisIn4
15	AnalogOut4	16	AnalogOut5
17	AnalogOut6	18	AGND
19	AxisOut1	20	AxisOut2
21	AxisOut3	22	AxisOut4
23	GND	24	AGND
25	AnalogOut7	26	AnalogOut8

3.4.5 Cable-5005-01

PMD Part #: Cable-5005-01

Description: Male to female straight through high density DB-15 connector cable.

Length: 1.0 meter

Cable: 15 conductor, with 26AWG stranded cables.

J16-J19 Pin	Signal	J16-J19 Pin	Signal
1	QuadA+	2	QuadA-
3	QuadB+	4	QuadB-
5	GND	6	Index+
7	Index-	8	HallA
9	HallB	10	HallC
11	Home	12	PosLim
13	NegLim	14	+5V
15	GND		

3.4.6 Cable-1007-01

PMD Part #: Cable-1007-01

Description: Male DB-9 to female DB-9 cable

Length: 1.5 meter

Cable: 3 conductor, 26AWG, Untwisted, shielded, UL STYLE 2464

J23 Pin	Signal	Connects to
1	RS485Select	-
2	Srl1Xmt/RS485Xmt-	2
3	Srl2Rcv/RS485Rcv+	3
4	No connect	-
5	GND	5
6	Srl2Rcv/RS485Rcv+	6
7	Srl1Rcv/ RS485Rcv-	7
8	Srl2Xmt/RS485Xmt+	8
9	Srl1Xmt/RS485Xmt-	9

3.4.7 Cable-RJ45-02-R

PMD Part #: Cable-RJ45-02-R

Description: Male RJ-45 to male RJ-45 cable wired in a straight-through configuration

Length: 2.0 meters

Cable: 4P, 24AWG, UTP, Cat 5e

CANbus

J21,J22 Pin	Signal	Connects to
1	CAN+	1
2	CAN-	2
3	GND	3
4	pass thru	4
5	pass thru	5
6	pass thru	6
7	GND	7
8	pass thru	

Ethernet

J24 Pin	Signal	Connects to
1	EthernetTx+	1
2	EthernetTx-	2
3	EthernetRx+	3
4	-	4
5	-	5
6	EthernetRX-	6
7	-	7
8	-	8

3.5 Absolute Maximum Ratings

HV voltage range:	0V to +60V
+5V voltage range:	-0.3V to +5.5V
Storage Temperature:	-20 to +150 C

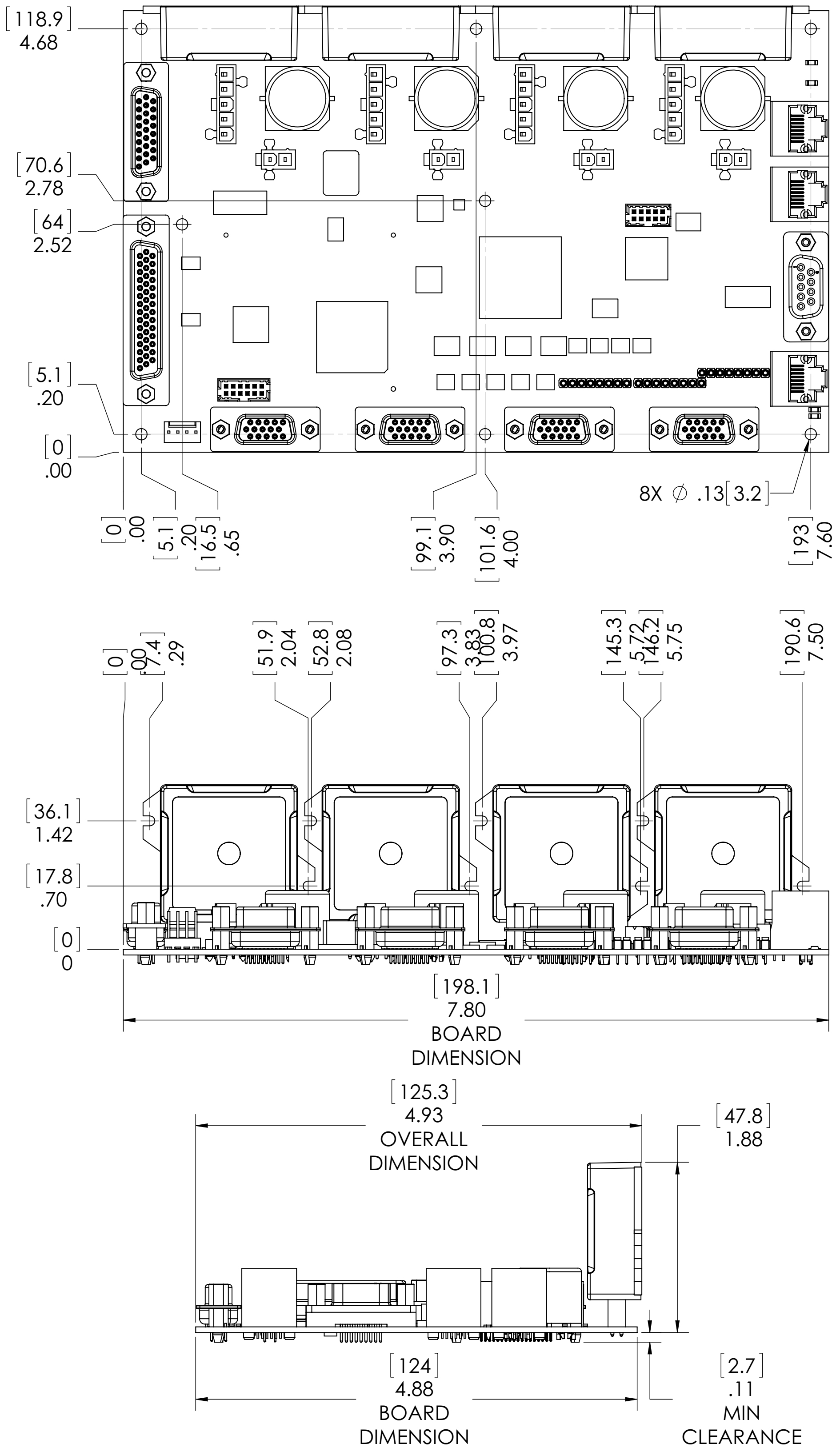
3.6 Environmental and Electrical Ratings

Storage temperature:	-40 to +125 degrees C (-40° F to +257° F)
Operating temperature:	0 to +70 degrees C (32° F to +158° F)
HV power requirement:	+12V to + 56V operating range
Optional +5V requirement:	4.75V to 5.25V operating range
Analog (DAC) output range:	-10.0V to +10.0V, ± 3mA min/axis, short circuit protected
Analog input range:	-10.0V to +10.0V, 80.4 KOhm input impedance
Digital I/O voltage range:	0V to 5V, TTL thresholds, inputs pulled up to 5V through 4.7 kOhm resistors
Digital outputs drive capacity:	DC output source or sink current: ± 50mA
CAN communications:	2.0B compliant, non-isolated, 1 Mbps
Serial communications:	RS232 signaling or RS485 Full or Half Duplex (data only)
Ethernet communications:	10/100BASE-TX (10/100 Mbps)

3.7 Mechanical Dimensions

The size of the board is 7.80" L x 4.88" W.

Figure 3-2: Prodigy/CME Machine-Controller Mechanical Dimensions



3.8 L-Bracket Mechanical Dimensions

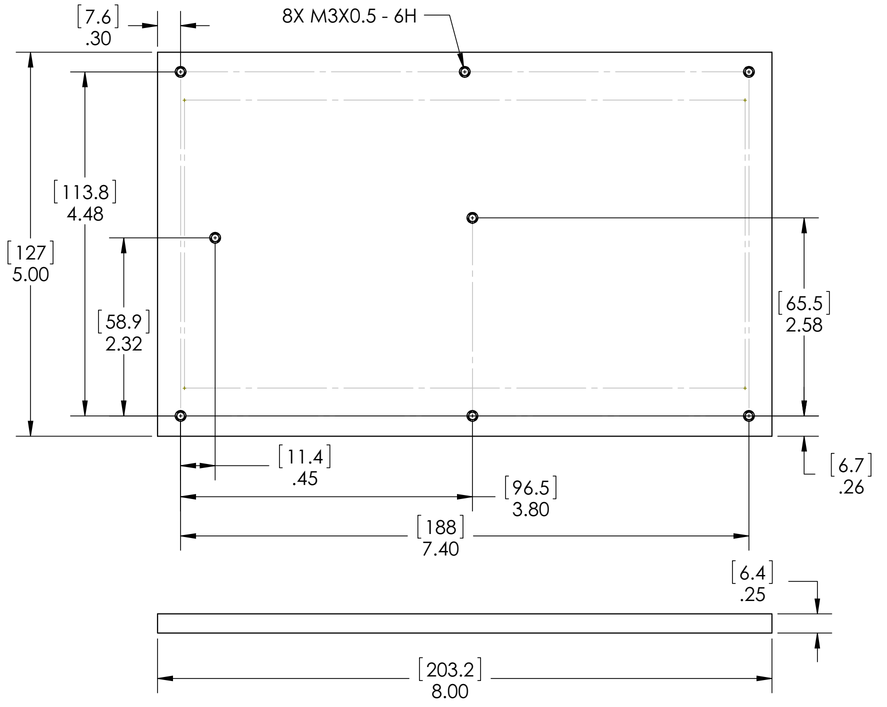


Figure 3-3: L-Bracket Base Mechanical Dimensions

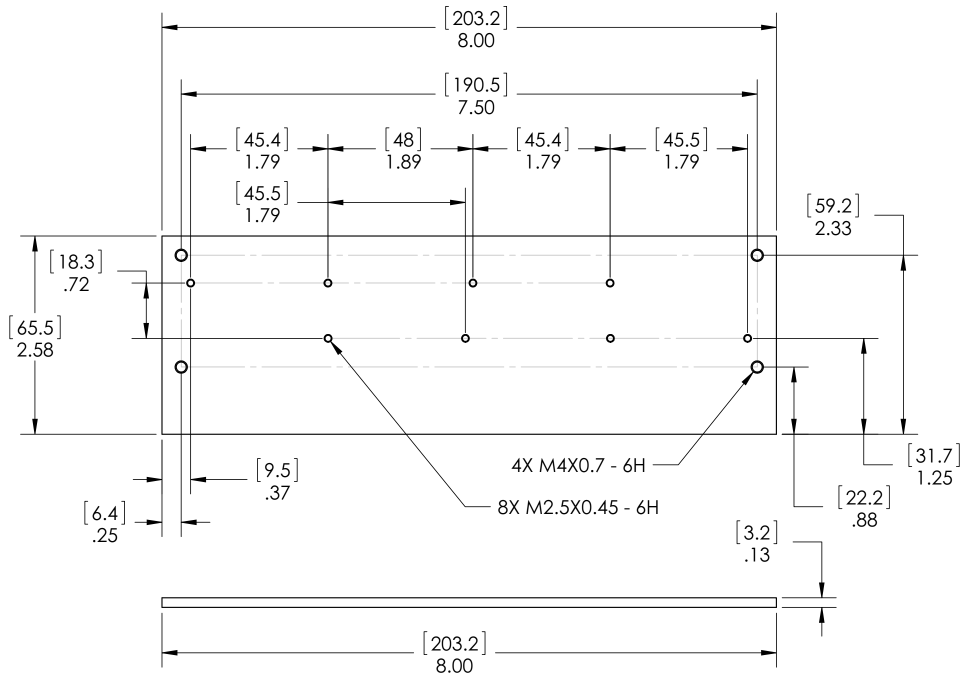
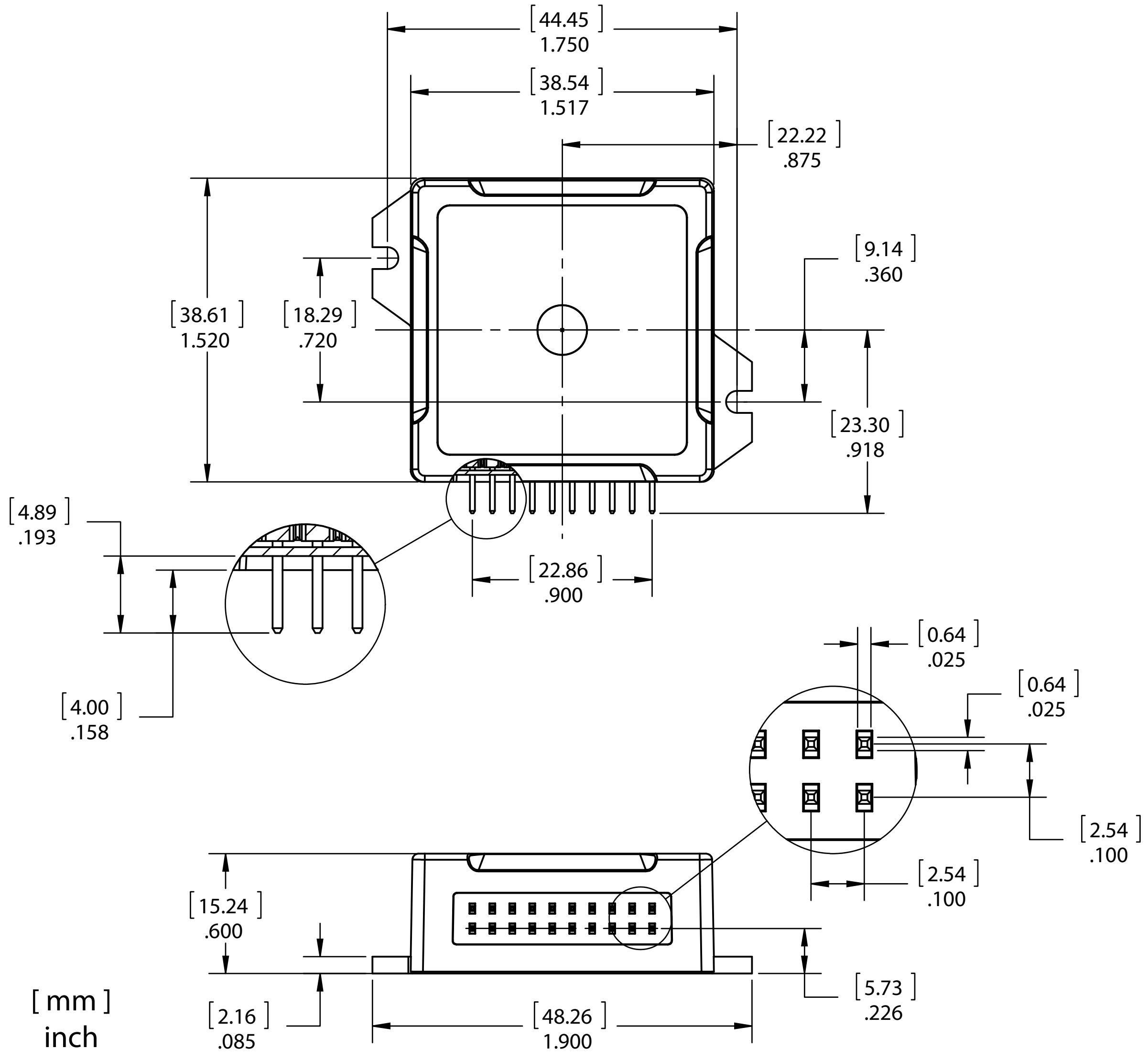


Figure 3-4: L-Bracket Vertical Plate Mechanical Dimensions

3.9 Atlas Mechanical Dimensions

Figure 3-5: Atlas Digital Amplifier - Vertical Unit with Tabs Mechanical Dimensions



Index