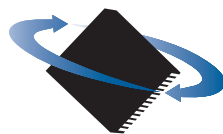


# Prodigy<sup>®</sup> /CME Machine-Controller User Guide



**PERFORMANCE  
MOTION DEVICES**

Performance Motion Devices, Inc.  
1 Technology Park Drive  
Westford, MA 01886



---

## NOTICE

This document contains proprietary and confidential information of Performance Motion Devices, Inc., and is protected by federal copyright law. The contents of this document may not be disclosed to third parties, translated, copied, or duplicated in any form, in whole or in part, without the express written permission of Performance Motion Devices, Inc..

The information contained in this document is subject to change without notice. No part of this document may be reproduced or transmitted in any form, by any means, electronic or mechanical, for any purpose, without the express written permission of Performance Motion Devices, Inc..

Copyright 1998–2020 by Performance Motion Devices, Inc.

Juno, ATLAS, Magellan, ION, Prodigy, Pro-Motion, C-Motion, and VB-Motion are registered trademarks of Performance Motion Devices, Inc.

## Warranty

Performance Motion Devices, Inc. warrants that its products shall substantially comply with the specifications applicable at the time of sale, provided that this warranty does not extend to any use of any Performance Motion Devices, Inc. product in an Unauthorized Application (as defined below). Except as specifically provided in this paragraph, each Performance Motion Devices, Inc. product is provided “as is” and without warranty of any type, including without limitation implied warranties of merchantability and fitness for any particular purpose.

Performance Motion Devices, Inc. reserves the right to modify its products, and to discontinue any product or service, without notice and advises customers to obtain the latest version of relevant information (including without limitation product specifications) before placing orders to verify the performance capabilities of the products being purchased. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgment, including those pertaining to warranty, patent infringement and limitation of liability.

## Unauthorized Applications

Performance Motion Devices, Inc. products are not designed, approved or warranted for use in any application where failure of the Performance Motion Devices, Inc. product could result in death, personal injury or significant property or environmental damage (each, an “Unauthorized Application”). By way of example and not limitation, a life support system, an aircraft control system and a motor vehicle control system would all be considered “Unauthorized Applications” and use of a Performance Motion Devices, Inc. product in such a system would not be warranted or approved by Performance Motion Devices, Inc..

By using any Performance Motion Devices, Inc. product in connection with an Unauthorized Application, the customer agrees to defend, indemnify and hold harmless Performance Motion Devices, Inc., its officers, directors, employees and agents, from and against any and all claims, losses, liabilities, damages, costs and expenses, including without limitation reasonable attorneys’ fees, (collectively, “Damages”) arising out of or relating to such use, including without limitation any Damages arising out of the failure of the Performance Motion Devices, Inc. product to conform to specifications.

In order to minimize risks associated with the customer’s applications, adequate design and operating safeguards must be provided by the customer to minimize inherent procedural hazards.

## Disclaimer

Performance Motion Devices, Inc. assumes no liability for applications assistance or customer product design. Performance Motion Devices, Inc. does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of Performance Motion Devices, Inc. covering or relating to any combination, machine, or process in which such products or services might be or are used. Performance Motion Devices, Inc.’s publication of information regarding any third party’s products or services does not constitute Performance Motion Devices, Inc.’s approval, warranty or endorsement thereof.

## Patents

Performance Motion Devices, Inc. may have patents or pending patent applications, trademarks, copyrights, or other intellectual property rights that relate to the presented subject matter. The furnishing of documents and other materials and information does not provide any license, express or implied, by estoppel or otherwise, to any such patents, trademarks, copyrights, or other intellectual property rights.

Patents and/or pending patent applications of Performance Motion Devices, Inc. are listed at <https://www.pmdcorp.com/company/patents>.

## Related Documents

### **Magellan® Motion Control IC User Guide**

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

### **Atlas Digital Amplifier User 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 Control IC Programming Reference**

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

### **PMD Resource Access Protocol Programming Reference**

Descriptions of all Prodigy and ION 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.

# Table of Contents

---

<b>Chapter 1. Installation</b>	<b>9</b>
1.1 Prodigy/CME Machine-Controller Overview	9
1.2 Machine-Controller Developer Kits	10
1.3 Installation Overview	12
1.4 Recommended Hardware	13
1.5 Developer Kit Assembly	14
1.6 Software Installation	17
1.7 Preparing the Board for Installation	19
1.8 Connection Summary	20
1.9 Applying Power	25
1.10 First-Time System Verification	25
1.11 Developing User Application Code	31
<b>Chapter 2. Operation</b>	<b>35</b>
2.1 Board Function Summary	36
2.2 Magellan Functions	38
2.3 I/O Functions	49
2.4 Communications Functions	56
2.5 Atlas Amplifier Functions	60
2.6 General Board Functions	64
2.7 C-Motion Engine Functions	70
2.8 Software Libraries	74
<b>Chapter 3. Electrical Reference</b>	<b>75</b>
3.1 User-Settable Components	75
3.2 Connectors	77
3.3 Motor Connections	86
3.4 Cables	88
3.5 Absolute Maximum Ratings	90
3.6 Environmental and Electrical Ratings	91
3.7 Certifications & Compliance	91
3.8 Mechanical Dimensions	92
3.9 L-Bracket Mechanical Dimensions	93
<b>Index</b>	<b>95</b>



---

This page intentionally left blank.

# List of Figures

1-1	Developer Kit Elements .....	10
1-2	Mounting Machine Controller Board onto L-Bracket Base Plate .....	14
1-3	Thermal Transfer Material Attachment .....	15
1-4	Atlas Installation into Machine-Controller Board .....	15
1-5	Compact to Ultra Compact Atlas Format Converter .....	16
1-6	Attaching Atlas Units to Vertical Plate .....	17
1-7	Prodigy/CME Machine-Controller Board Component Location .....	19
1-8	With Brushless DC Motor .....	20
1-9	With DC Brush Motor .....	21
1-10	With 2-Phase Step Motor .....	21
1-11	Power Connections .....	23
1-12	Serial Port Connection .....	24
1-13	Serial and Ethernet Connection .....	25
1-14	Two Ways to Locate the Code on the Prodigy/CME Machine-Controller Board .....	32
2-1	Prodigy/CME Machine-Controller Board Internal Block Diagram .....	35
2-2	Encoder Signal Schematic QuadA/B and Index .....	41
2-3	SSI with Controller as Clock Master .....	43
2-4	Connecting Multiple Boards .....	43
2-5	Home, Limits, and Hall Sensor Signal Schematic .....	45
2-6	AxisIn Signal Schematic .....	46
2-7	AxisOut Signal Schematic .....	46
2-8	Analog Output Signal Schematic .....	48
2-9	DigitalIn(1-4) Interface Schematic .....	52
2-10	DigitalOut(1-4) Interface Schematic .....	52
2-11	DigitalIO(1-8) Interface Schematic .....	53
2-12	Analog Inputs Simplified Diagrams .....	54
2-13	Analog Input Interface .....	55
2-14	Analog Output Interface .....	56
2-15	Serial Interface Schematic .....	58
2-16	CANbus Interface Schematic .....	59
2-17	Current Foldback Processing Example .....	63
2-18	On-board Dual-ported Memory .....	64
2-19	AmpEnable Output Interfacing .....	67
2-20	Overview of C-Motion Engine Architecture .....	71
3-1	Components and Layout, Front of Board .....	75
3-2	Prodigy/CME Machine-Controller Mechanical Dimensions .....	92
3-3	L-Bracket Base Mechanical Dimensions .....	93
3-4	L-Bracket Vertical Plate Mechanical Dimensions .....	94

This page intentionally left blank.



# 1. Installation

1

## In This Chapter

- ▶ Prodigy/CME Machine-Controller Overview
- ▶ Machine-Controller Developer Kit
- ▶ Installation Overview
- ▶ Recommended Hardware
- ▶ Developer Kit Assembly
- ▶ Software Installation
- ▶ Preparing Board for Installation
- ▶ Connection Summary
- ▶ Applying Power
- ▶ First-Time System Verification
- ▶ Developing User Application Code

## 1.1 Prodigy/CME Machine-Controller Overview

This manual provides a complete user guide for the Prodigy/CME Machine-Controllers. For documentation on other members of the Prodigy family please consult the appropriate documentation.

The Prodigy/CME Machine-Controller boards provide high-performance control of DC Brush, Brushless DC, and step motors. All Prodigy boards are based on PMD's Magellan<sup>®</sup> Motion Control ICs, which perform high speed motion control functions such as profile generation, servo loop closure, pulse & direction signal generation, and many other real-time functions.

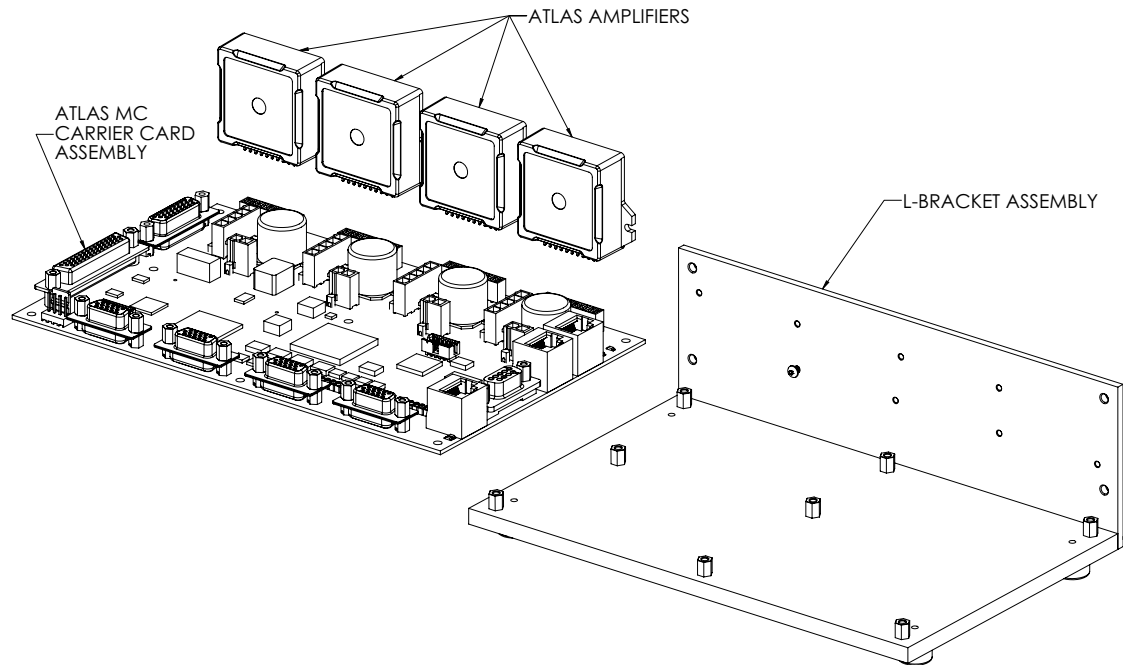
The Prodigy/CME machine-controller boards include a C-Motion<sup>®</sup> Engine (CME), which allows the user to offload application code from the host onto the motion board. In addition, machine controller boards provide the option of on-board amplification via PMD's Atlas Amplifiers.

Prodigy/CME Machine-Controllers come in various configurations of axis number from 1 to 4 as shown in the following table:

<b>Prodigy/CME Machine-Controller P/N</b>	<b>Installed Magellan P/N</b>	<b>Number of Axes</b>	<b>Motor Type</b>
PR33SM00001	MC58120	1	Brushless DC, DC Brush, Step Motor
PR33SM00002	MC58220	2	Brushless DC, DC Brush, Step Motor
PR33SM00003	MC58320	3	Brushless DC, DC Brush, Step Motor
PR33SM00004	MC58420	4	Brushless DC, DC Brush, Step Motor

## 1.2 Machine-Controller Developer Kits

**Figure 1-1:  
Developer Kit  
Elements**



To facilitate initial system development and integration PMD offers developer kits for Prodigy/CME Machine-Controllers. A typical machine controller developer kit setup consists of five elements:

- 1-4 axis machine controller board
- L-bracket mounting hardware consisting of a base plate with PCB mounting posts attached to a vertical plate for Atlas mounting (optional if no Atlas units installed)
- Up to 4 Atlas Digital Amplifier units
- Miscellaneous cables & accessories to facilitate connection to the application motor hardware and to provide various other electrical connections
- Software SDK (software developer kit) and PDFs for all Prodigy documentation

### 1.2.1 Developer Kit Part Numbers

There are several available machine controller developer kits, designed to address a range of applications and machine controller configurations. Depending on the developer kit P/N ordered some assembly of the developer kit may be required by the user, or the developer kit may come completely assembled and ready to use out of the box.

The following table provides detailed information on each available machine controller developer kit P/N:

Developer Kit P/N	Items Included	Comments
PRK33MK00004	L-Bracket Cables & accessories Software SDK & manuals	Some assembly required. This P/N is typically ordered when the board and Atlas units will be purchased separately.
PRK33ML44002	Machine Controller board (2 axis) 2 high power multi motor Atlas units L-Bracket Cables & Accessories Software SDK & manuals	Comes fully assembled. This is a complete 2-axis setup with Atlas units and machine controller board included.
PRK33ML44402	Machine Controller board (3 axis) 3 high power multi motor Atlas units L-Bracket Cables & Accessories Software SDK & manuals	Comes fully assembled. This is a complete 3-axis setup with Atlas units and machine controller board included.
PRK33ML44442	Machine Controller board (4 axis) 4 high power multi motor Atlas units L-Bracket Cables Accessories Software SDK & manuals	Comes fully assembled. This is a complete 4-axis setup with Atlas units and machine controller board included.

For more information on Atlas amplifier configurations refer to the *Atlas Digital Amplifier User Manual*.

The PRK33MK00004 Developer Kit consists of the mounting and connection hardware only. To create a complete functioning setup using this DK part, the machine controller board and Atlas units must be ordered separately.



## 1.2.2 L-Bracket

As shown in [Figure 1-1](#) an L-bracket assembly is part of the Prodigy/CME Machine-Controller Developer Kit. 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 Atlas amplifiers

For complete mechanical drawings of the L-bracket component see [Section 3.9, “L-Bracket Mechanical Dimensions.”](#)

## 1.2.3 Developer Kit Cables & Accessories

All Prodigy/CME Machine-Controller Developer Kits come with the following cables and accessories:

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 board's CANbus connector and has RJ45 connectors on both ends.

Component Part Number	Description
Cable-RJ45-02-R	Ethernet connector. This cable connects to the board's Ethernet connector, and has RJ45 connectors on both ends.
MC-HW-03	Breakout interconnect module and cable provides convenient jack-screw type terminators for the 15-pin Axis Connectors.
MC-HW-04	Breakout interconnect module and cable provides convenient jack-screw type terminators for the 26-pin Amplifier I/O Connector.
MC-HW-05	Breakout interconnect module and cable provides convenient jack-screw type terminators for the 44-pin General I/O Connector.
Cable-5001-01	Molex 2-circuit stub cable
Cable-5002-01	Moles 5-circuit stub cable

### 1.2.4 Optional Accessories

The following table shows components that are not included with the developer kit but that you may find useful for your development. To purchase these components please contact your local PMD representative:

Component Part Number	Description
Adapt-USB232-01.R	This adaptor provides USB to serial conversion. It is useful for connecting to the Prodigy/CME's D8-9 serial port from a USB port

## 1.3 Installation Overview

- 1 If you ordered a non-fully assembled version of the machine controller developer kit you will need to assemble the DK hardware. See [Section 1.5, “Developer Kit Assembly,”](#) for complete assembly instructions. If you purchased one of the pre-assembled DKs, then no assembly is required.
- 2 Before using the board, the software must be installed. See [Section 1.6, “Software Installation,”](#) for instructions on installing the software.
- 3 For a normal installation of a Prodigy/CME Machine-Controller board you will need to configure the board for the specific motion hardware to which it will be connected. See [Section 1.7, “Preparing the Board for Installation,”](#) for a description of configuring the boards.
- 4 Next, connect the system's motors, encoders, amplifiers, and sensors to operate the motion hardware. See [Section 1.8, “Connection Summary,”](#) for a description of the available connections and options for the Prodigy/CME Machine-Controller board.
- 5 Connect the Prodigy/CME Machine-Controller board 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 1.8, “Connection Summary,”](#) for a description of the communications connections and options.
- 6 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 1.10, “First-Time System Verification,”](#) for a description of this procedure.

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

## 1.4 Recommended Hardware

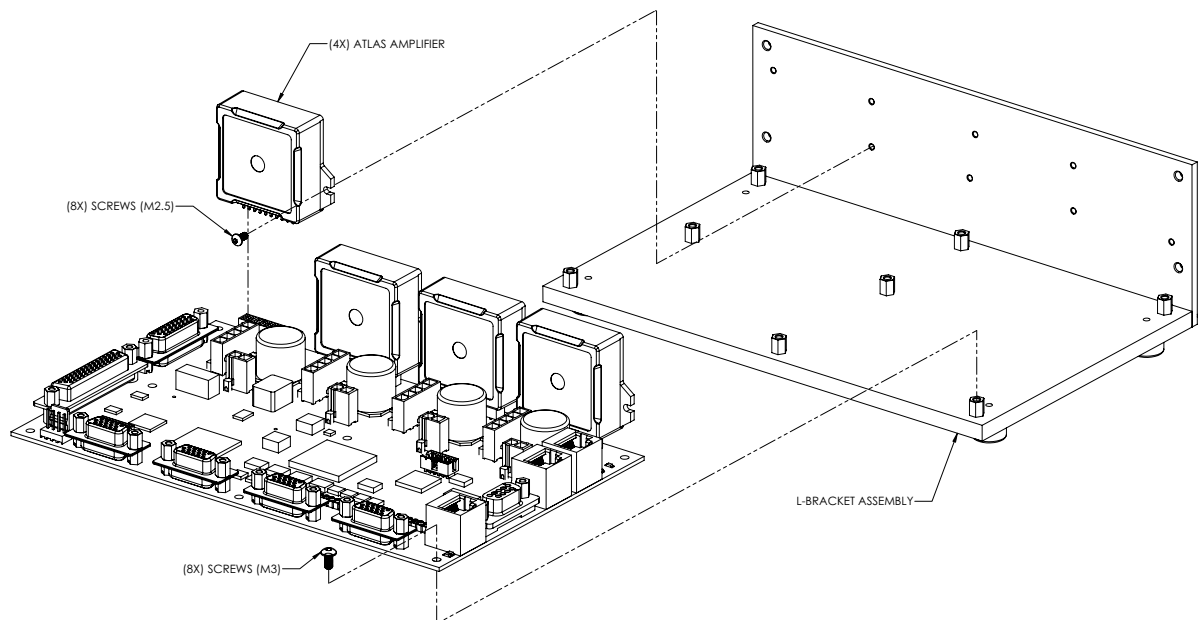
To install a Prodigy/CME Machine-Controller board the following hardware is recommended.

- Intel (or compatible) processor, 1 Gbyte of available disk space, 256 MB of available RAM, and a CD ROM drive. The supported PC operating systems are Windows XP, Vista, Windows 7, and Windows 8.
- 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 DK the Atlases may already be installed, or you may need to install them yourself. Note that the machine controller board can be operated without on-board Atlas amplifiers using off-board 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 board itself is powered from the Atlas power input for axis #1 using on-board DC to DC circuitry. For first-time installation you will connect to both the board's serial port and Ethernet port.

## 1.5 Developer Kit Assembly

### 1.5.1 Machine Controller Board Installation

**Figure 1-2:**  
Mounting  
Machine  
Controller  
Board onto  
L-Bracket Base  
Plate



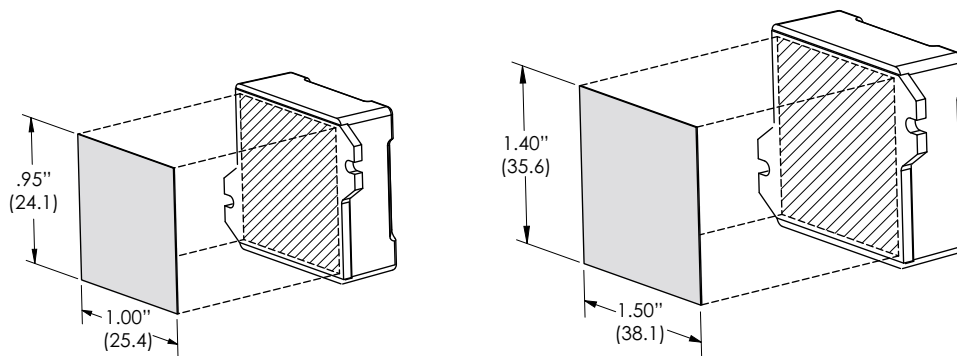
If you are not using an L-bracket, or if the developer kit you ordered already has the machine controller board installed onto the L-bracket hardware, you may skip to [Section 1.5.3, “Installing Atlas Units into the Board.”](#)

To install the machine controller board to the base plate use eight M3 screws at the base plate post location as shown in [Figure 1-2](#). All necessary fasteners should be included with your developer kit hardware. Also included are the Allen keys which you will need for installation of the machine controller and Atlas amplifiers.



It is good practice to wear a grounding strap while handling both the machine controller board and the Atlas units. In addition it is recommended that assembly be undertaken on a surface that dissipates electrostatic charge.

## 1.5.2 Atlas Thermal Pad Attachment



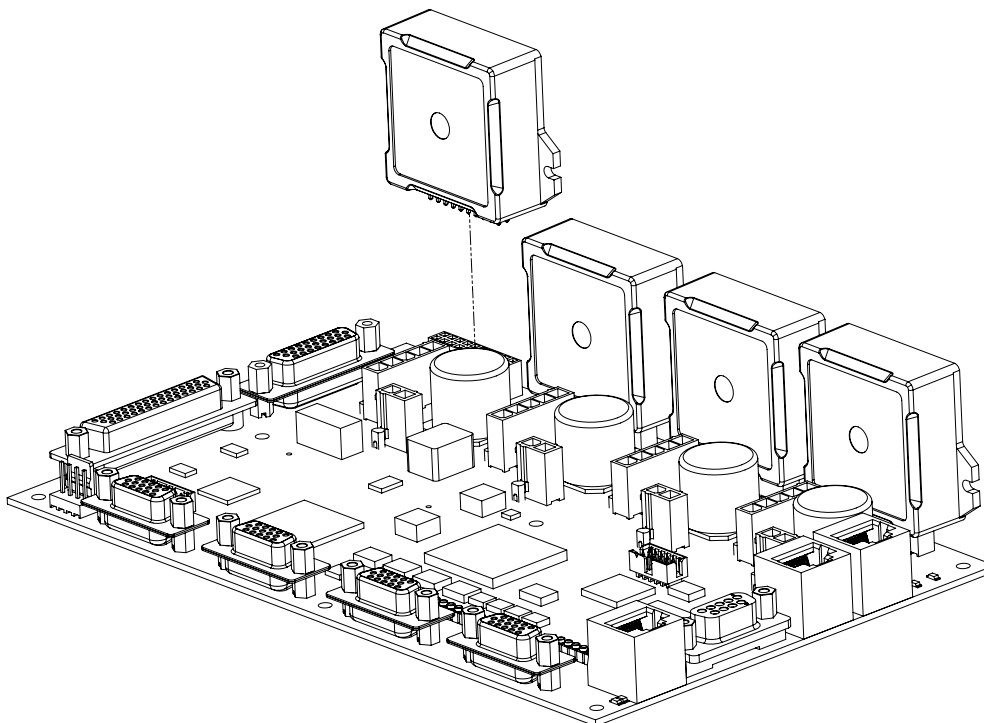
**Figure 1-3:**  
Thermal  
Transfer  
Material  
Attachment

It is very important to have good thermal contact between the Atlas units and the L-bracket heat sink. For this purpose thermal pads will be attached to each Atlas to be installed.

First, locate the thermal pads of matching size. Two sets of thermal pads are included in your developer kit reflecting the two available Atlas package sizes - compact or ultra-compact. As shown in the figure above compact units use the larger thermal pad and ultra compact units use the smaller thermal pad.

Next, carefully remove the thin plastic protective sheets on either side of each pre-cut thermal pad and mount onto the Atlas unit, carefully aligning the pads with the Atlas' metallic backing, and applying finger pressure to adhere the pads to the metal. Note that the dimensions of each pad are not exactly square, so it is best to align the pads in the orientation shown in the diagram. Once pressed in place the pads should stay in place, but if required the pads can be removed and remounted.

## 1.5.3 Installing Atlas Units into the Board



**Figure 1-4:**  
Atlas  
Installation into  
Machine-  
Controller  
Board

If your machine controller board already has the Atlas amplifiers installed you may skip to [Section 1.6, “Software Installation.”](#)

To install Atlas units into the machine controller board sockets, confirm that the Atlas is oriented correctly, with the metal heat sink surface facing toward the vertical L-bracket plate. Carefully align the Atlas pins to the socket and press firmly down until the Atlas is fully seated in the socket.

If using Atlas units for specific motor types, 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. In the figure above the axis #1 socket is the left-most socket and increase to the right. Alternatively by using the multi-motor version of Atlas, the motor type can be user programmed.

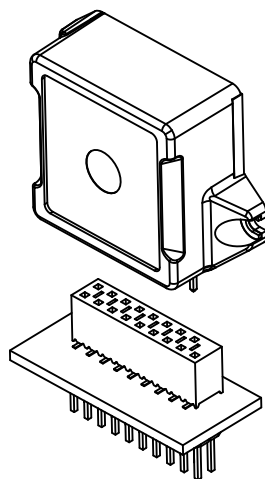


Extreme care should be taken when installing the Atlas into its socket. Failure to orient the Atlas correctly, or mis-alignment of pins may result in damage to the Prodigy/CME Machine-Controller board, Atlas units, or both.

### 1.5.3.1 Ultra Compact Atlas Unit Installation

The above instructions provide details on installing compact Atlas vertical units, which are the larger of the two available Atlas sizes. The smaller ultra compact units, which are the package sizes for the low and medium power Atlas units, require the installation of a conversion card before installation into the machine controller board.

**Figure 1-5:**  
Compact to  
Ultra Compact  
Atlas Format  
Converter



For any ultra compact Atlas units to be installed onto the machine controller board, first install the smaller thermal pads provided with your developer kit. Other than the size difference, installation of the pad to the Atlas unit is the same as for compact Atlas units.

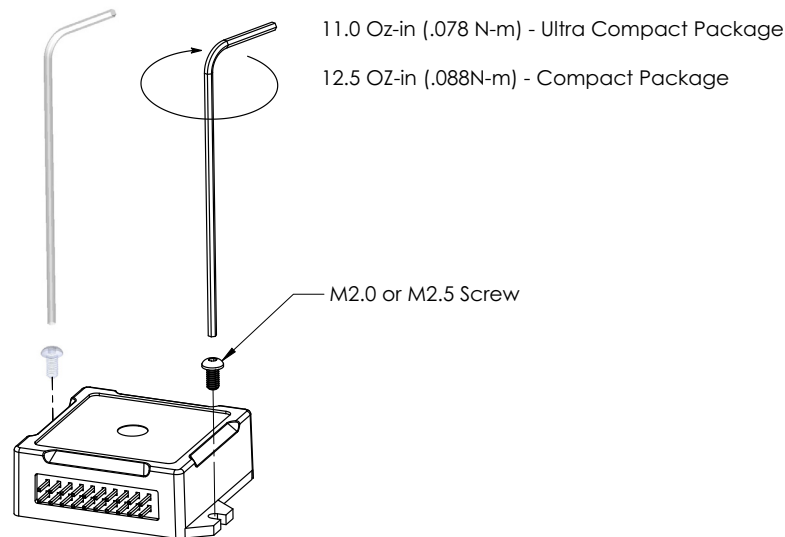
Next, each ultra-compact Atlas unit must be mated to a converter card as shown in the above figure. Before connecting the Atlas to the converter card, care should be taken to insure that they are oriented correctly, and that all pins align correctly without overhang.

Once the ultra-compact Atlas has been properly mated to the converter card, the converter/Atlas assembly can then be inserted into the machine controller board as described in [Section 1.5.3, “Installing Atlas Units into the Board.”](#)

Note that the socket installation location of the compact and ultra compact Atlas units on the machine controller board is interchangeable. There is no restriction on the location of compact Atlas units versus the location of ultra compact Atlas units.



## 1.5.4 Attaching Atlas Units to the Vertical Plate



**Figure 1-6:**  
Attaching Atlas  
Units to  
Vertical Plate

Finally, the Atlas units should be fastened to the vertical plate. Two screws are used to attach each Atlas and [Figure 1-6](#) shows how the screws connect to the vertical plate. For compact Atlas units (high power) the M2.5 screws are used, and for ultra compact Atlas units (low and medium power) M2 screws are used.

Note that the mounting tap hole locations in the vertical plate are different for the compact and ultra compact Atlas units. Use only modest force in attached Atlas units to the vertical plate. [Figure 1-6](#) shows this, also providing the torque limit specification for both Atlas types.

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

## 1.6 Software Installation

The software distribution for the Prodigy/CME Stand-Alone board developer kit is downloaded from the PMD website at the URL: <https://www.pmdcorp.com/resources/software>.

All software applications are designed to work with Microsoft Windows.

To install the software:

- 1 Go to the Software Downloads section of PMD's website located at <https://www.pmdcorp.com/resources/software> and select download for "Developer Kit Software"
- 2 After selecting download you will be prompted to register your DK, providing the serial # for the DK and other information about you and your motion application.
- 3 After selecting submit the next screen will provide a link to the software download. The software download is a zip file containing various installation programs. Select this link and downloading will begin.
- 4 Once the download is complete extract the zip file and execute the desired install programs from the list below. Every first-time installation should install Pro-Motion, and at least one of the two SDK options. However you may install both SDKs if desired. When installing the SDKs you will be given the option to download the documentation and/or the complete SDK content.

- Pro-Motion – an application for communicating to, and exercising PMD ICs, modules, or boards.
- PMD SDK – a software development kit for creating motion applications using the C/C++ programming languages. Also contains PDF versions\* of all PMD product documentation.
- CME SDK – a software development kit for creating motion applications using the .NET (C#, VB) programming languages. Also contains PDF versions\* of all PMD product documentation.

\*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>.

Here is more information on each of these software packages.

### 1.6.1 Pro-Motion

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

- Motion oscilloscope graphically displays processor parameters in real-time
- AxisWizard to automate axis setup and configuration
- Position loop and current loop auto-tuning
- Project window for accessing motion resources and connections
- Ability to save and load settings
- Distance, time, and electrical units conversion
- Frequency sweep and bode plot analysis tools
- 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

### 1.6.2 C-Motion

C-Motion provides a convenient set of callable routines comprising the C language code required for controlling Prodigy boards, whether running on a separate host computer such as a PC, on an embedded microcontroller, or running inside the Prodigy Board on the C-Motion Engine. C-Motion includes the following features:

- Magellan axis virtualization
- Ability to communicate to multiple PMD motion cards or modules
- Ability to communicate via PC/104 bus, serial, CANbus, Ethernet, SPI (Serial Peripheral Interface), or 8/16 bit parallel bus
- Provided as source code, allowing easy compilation & porting onto various run-time environments including a 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 Control IC Programming Reference*

## 1.6.3 .NET Language Support

A complete set of methods and properties is provided for developing applications in Visual Basic and C# 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, such as Labview, but no special software support is provided.

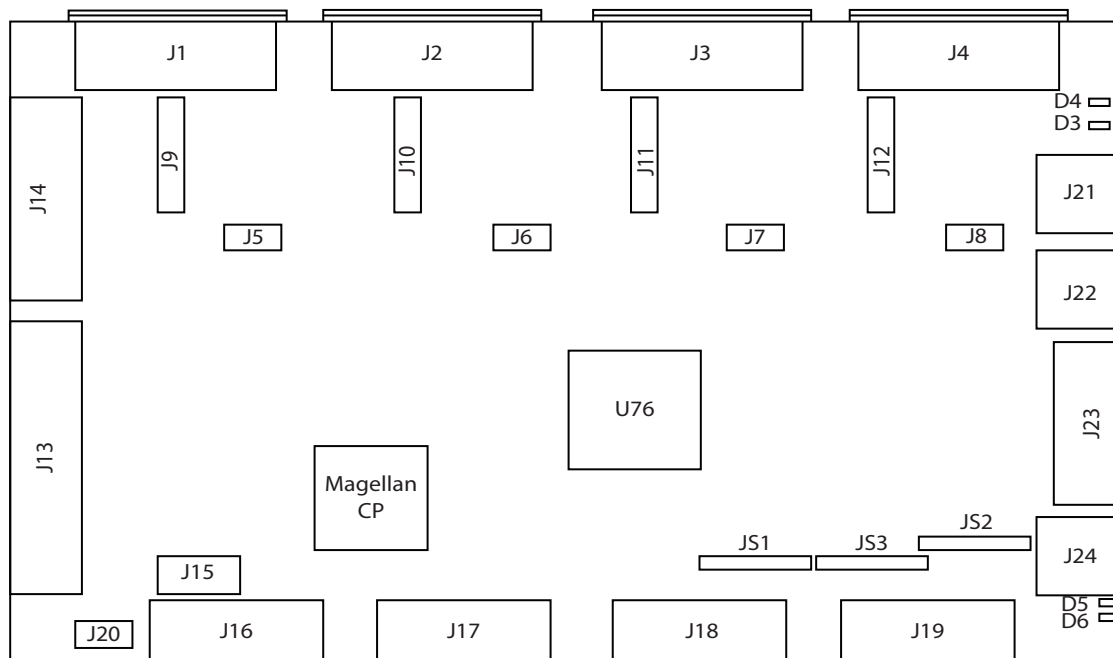
Includes the following features:

- Magellan axis virtualization
- Ability to communicate to multiple PMD motion cards or modules
- Ability to communicate via PC/104 bus, serial, CANbus, 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 Programming Reference*.

## 1.7 Preparing the Board for Installation

Figure 1-7 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 1-7:**  
Prodigy/CME  
Machine-  
Controller Board  
Component  
Location

### Board Components

The following table describes selected components on the board (as shown in Figure 1-7) and their functionality.

Label	Description
JS1-JS3	Resistor packs
J1, J2, J3, J4	Axis #1, 2, 3, and 4 Atlas sockets (respectively)

Label	Description
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

### 1.7.1 Resistor Pack Settings

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

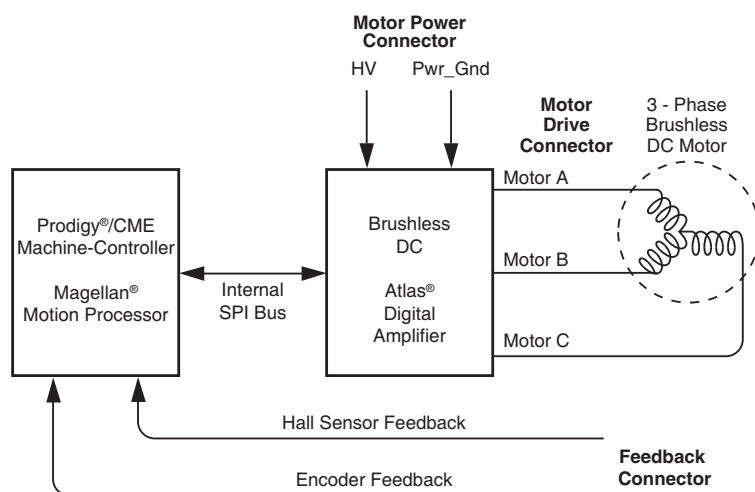
Item	Setting	Description
Resistor packs JS1, JS2, JS3	Installed; this is the default setting of resistor packs JS1–JS3.	If differential connections are being used, leave these resistor packs installed.
	Removed	If single-ended encoder connections are being used, remove the resistor packs.

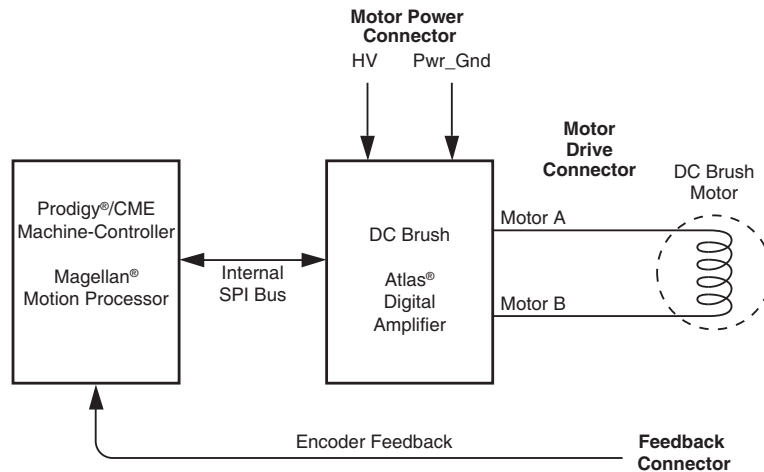
It is also possible to mix differential and single-ended encoder connections. Refer to [Section 3.1.1.3, “Using Both Single-Ended and Differential Encoder Connections,”](#) for details.

## 1.8 Connection Summary

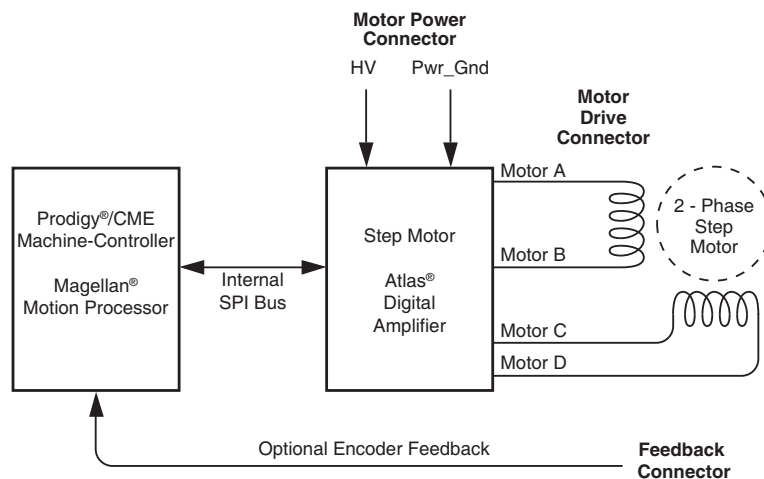
[Figure 1-8](#), [Figure 1-9](#), and [Figure 1-10](#) provide general connection overviews for each machine controller axis. For each axis there is a feedback connector, a motor drive connector, and a motor power connector.

**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 board with the attached motors. The following three sections detail how this should be done.

### 1.8.1 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 board 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 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. <i>(optional for step motor axes)</i>
2	QuadA-	Differential A- quadrature input. <i>(optional for step motor axes)</i>
3	QuadB+	Differential B+ quadrature input. <i>(optional for step motor axes)</i>
4	QuadB-	Differential B- quadrature input. <i>(optional for step motor axes)</i>

Pin #	Signal Name	Description
5	GND	This is the preferred ground connection for the quadrature and Index signal inputs
6	Index+	Differential Index+ quadrature input. <i>(optional for step motor axes)</i>
7	Index-	Differential Index- quadrature input. <i>(optional for step motor axes)</i>
8	HallA	Hall signal input phase A. <i>not used for DC brush or step motors</i>
9	HallB	Hall signal input phase B. <i>not used for DC brush or step motors</i>
10	HallC	Hall signal input phase C. <i>not used for DC brush or step motors</i>
11	Home	Home signal input <i>(optional)</i>
12	PosLim	Positive position limit input <i>(optional)</i>
13	NegLim	Negative position limit input <i>(optional)</i>
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 board using [Figure 1-7](#).

## 1.8.2 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 board 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 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:

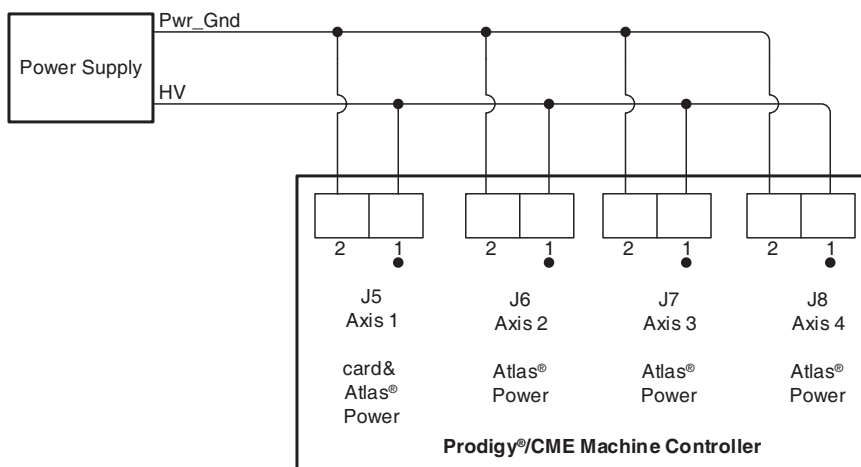
Motor type	Machine Controller Motor Lead	Motor Coil Connections
Brushless DC	Motor A	A winding connection
	Motor B	B winding connection
	Motor C	C winding connection
	Shield	(optional) motor shield connection
DC Brush	Motor A	+ winding connection
	Motor B	- winding connection
	Shield	(optional) motor 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 board using [Figure 1-7](#).

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).



### 1.8.3 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 board logic power is derived using on-board DC-DC converters. Note that under some circumstances it may be desirable to provide just the +5V board 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 board 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 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 board using Figure 1-5.

See [Chapter 3, Electrical Reference](#), for complete machine controller electrical specifications.

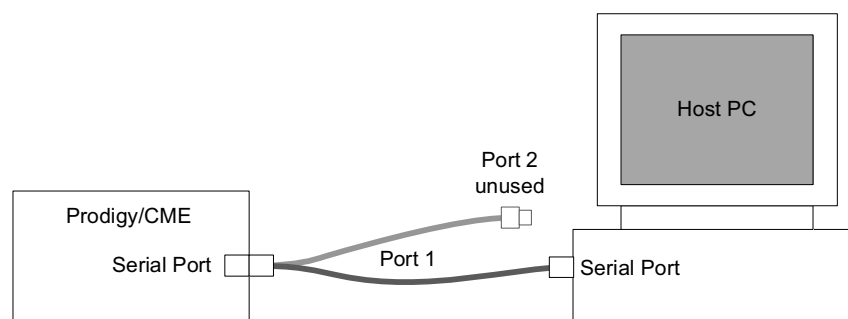
## 1.8.4 Communication Connections

While the Prodigy/CME Machine-Controller board 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 board for Ethernet communications. To set up the board for operation in other communication modes, see [Chapter 2, Operation](#).

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

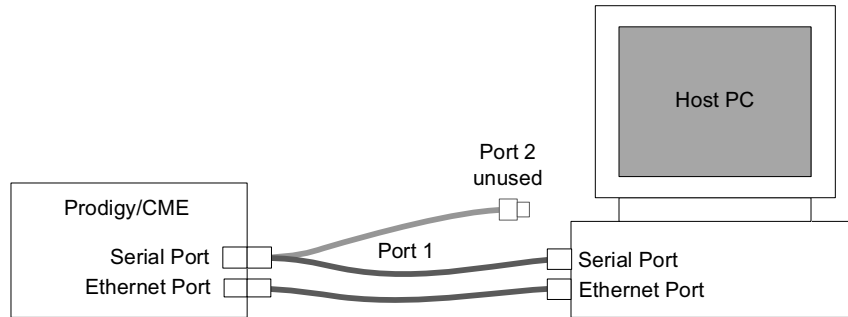
**Figure 1-12:  
Serial Port  
Connection**



The Ethernet connection will not be made until serial communications are established. See [Section 1.10.2, “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 board using [Figure 1-13](#).





**Figure 1-13:**  
Serial and  
Ethernet  
Connection

## 1.9 Applying Power

Once you have made your motion hardware, communication, and power connections, hardware installation is complete and the board 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 1-7](#). 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 board and check the motor and encoder connections. If anomalous behavior is still observed, call PMD for assistance.

## 1.10 First-Time System Verification

The first time system verification procedure summarized below has two overall goals. The first is to connect the Prodigy board 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 boards 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 board using the serial communications link.
- 2 Use the serial communications link to change the Ethernet communications parameters of the Prodigy/CME Machine-Controller board.
- 3 Establish communication between the PC and the board 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.

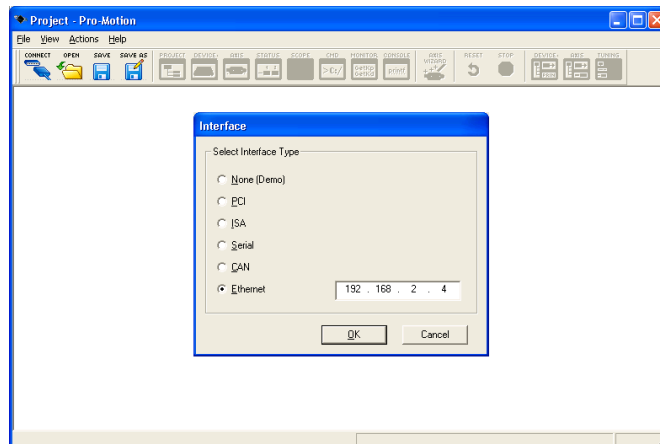
During this first time system setup you may find it useful to refer to other PMD manuals including the *Magellan Motion Control IC User Guide* to familiarize yourself with operation of the Magellan Motion Control IC, which lies at the heart of all Prodigy Motion boards.

## 1.10.1 Establishing Serial Communications

To establish serial communications:

- 1 Make sure the Prodigy board 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.



- 3 Click the Connect icon on the toolbar.

Alternatively, on the File menu, click Connect.

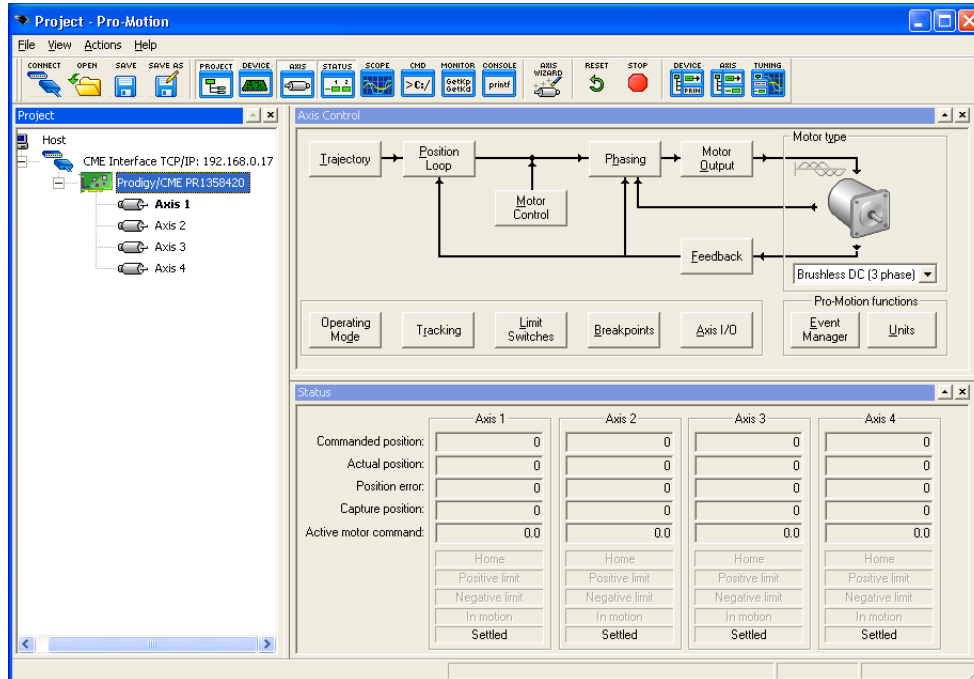
The purpose of the Interface dialog box is to indicate to Pro-Motion how your Prodigy/CME Machine-Controller board is connected to the PC. It provides various selectable communication options such as PCI, serial, CANbus, Ethernet.

- 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.



For example for a four axis Prodigy board, you see the board name next to an icon of a board, and below that you see four axis icons, one for each available axis of the motion board. Highlighting (single clicking) either the board icon or one of the axis icons with the mouse is used to select specific boards 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.

## 1.10.2 Setting the Ethernet Parameters

To set the Ethernet parameters:

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

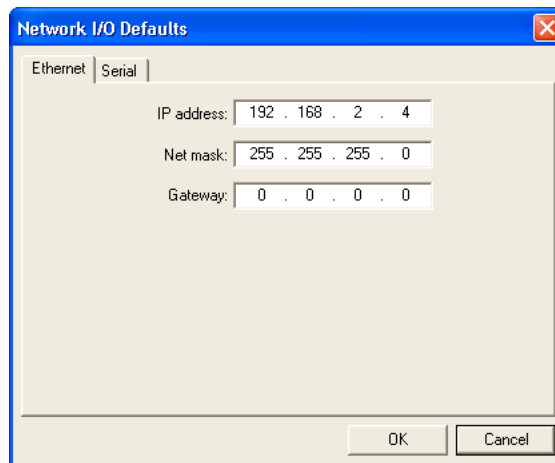
The Device window appears.

- 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.



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

- 7 Connect the Ethernet cable.

See [Section 1.8.4, “Communication Connections,”](#) for details.

The board is now ready for Ethernet communications.

### 1.10.3 Establishing Ethernet Communications

The board'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 board. 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 board.
- 4 When complete, click OK.

If Ethernet communications are successful, an additional set of graphical icons representing your board 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 1.10.2, “Setting the Ethernet Parameters,”](#) for details.

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

## 1.10.4 Disconnecting Serial Communications

To disconnect serial communications:

- 1 Select the serial link version of the Prodigy/CME Machine-Controller board 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 board icon and axes graphical icons in the Project box disappear, leaving only the Ethernet link icons for the board 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 board 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 board while another PC provides commands to that same board. Be aware however that two or more users sending motion commands to the same board can cause unexpected motion, and should be avoided.



When connecting your Prodigy board for use on an Ethernet network, be sure that the IP address provided for the Prodigy board does not conflict with the addresses of other users on the network. See [Section 1.10.2, “Setting the Ethernet Parameters,”](#) for a description of changing the IP address.

## 1.10.5 Initializing Motion Axes

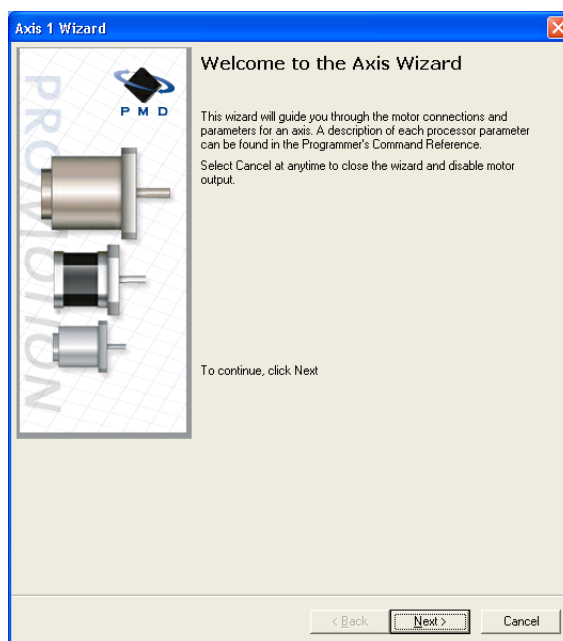
The next step to verify the correct operation of the system is to initialize each axis of the motion system sequentially, thereby verifying correct amplifier connection, encoder feedback connections (if an encoder is used), and other motion functions. All of this can be conveniently accomplished using 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.



- 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. 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 Control IC. Should this happen, it is possible to perform a manual tuning or commutation setup if desired.

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.

### 1.10.6 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:

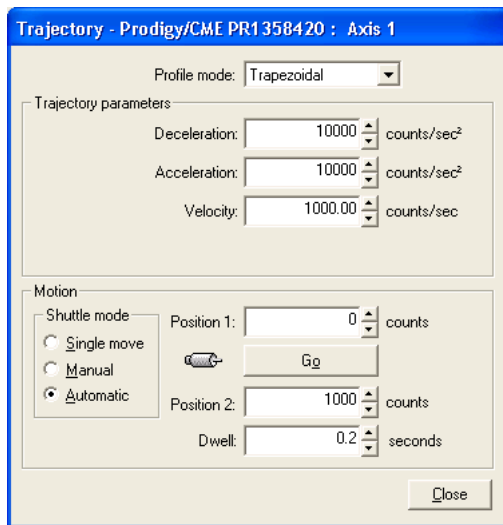
- 1 In the Project Window, select the motion axis that you would like to move by clicking the corresponding icon.
- 2 Click the Axis view button on the far right of the toolbar.

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.

- Click the Trajectory button in the Axis Control window

The Trajectory dialog box appears.



- In the Profile mode list, select Trapezoidal.
- 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<sup>2</sup>. 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<sup>2</sup>

Acceleration: 5,000 counts/sec<sup>2</sup>

Velocity: 10,000 counts/sec

Position 1: 0 counts

Position 2: 50,000 counts

- 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 1.10.5, “Initializing Motion Axes,”](#) and repeat the steps.



## 1.11 Developing User Application Code

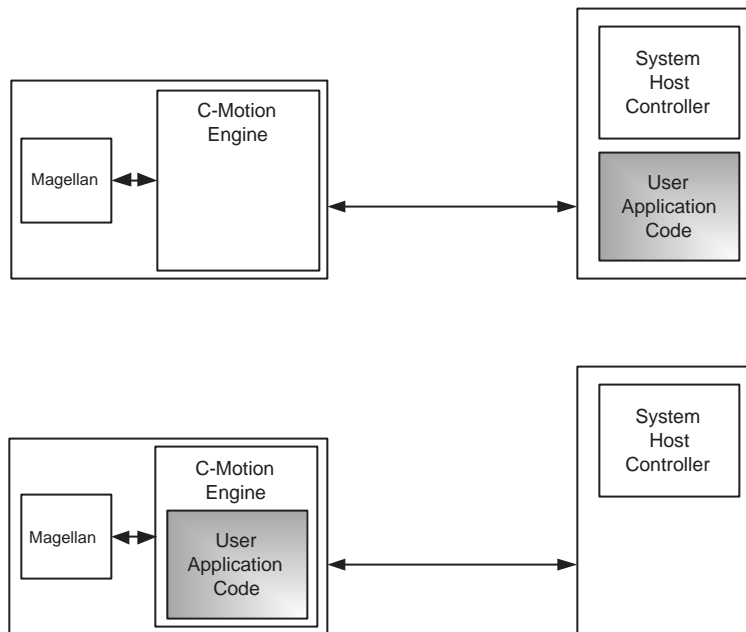
Pro-Motion provides an intuitive, convenient, graphical interface to setup and exercise motion controllers such as Prodigy/CME boards or ION® Digital Drives. Eventually though, you will begin to design application-specific software code that will serve as your system controller. PMD supports two standard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

For more information on Visual Basic and C# support or C-Motion, please consult the *Magellan Motion Control IC Programming Reference* and the *PMD Resource Access Protocol Programming Reference*.

### 1.11.1 Architecture

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

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



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

Either Visual Basic and C# or C-Motion can be used to communicate to the Prodigy/CME board, 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 board, the User code communicates directly to the resources available on the board such as the Magellan Motion Control IC. This has speed advantages both in communicating with those resources, and in real time code execution predictability.

Another feature of locating code on the board 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 application code, downloaded onto the board, forms a local controller while still communicating to the higher level controller that synchronizes overall system behavior.

Finally locating the code in the board allows use of the Prodigy/CME board as a fully standalone controller. In this mode a host controller network communication link is not used, and one or more of the board'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 board, 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.



## 1.11.2 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 board 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 board.

This page intentionally left blank.

## 2. Operation

2

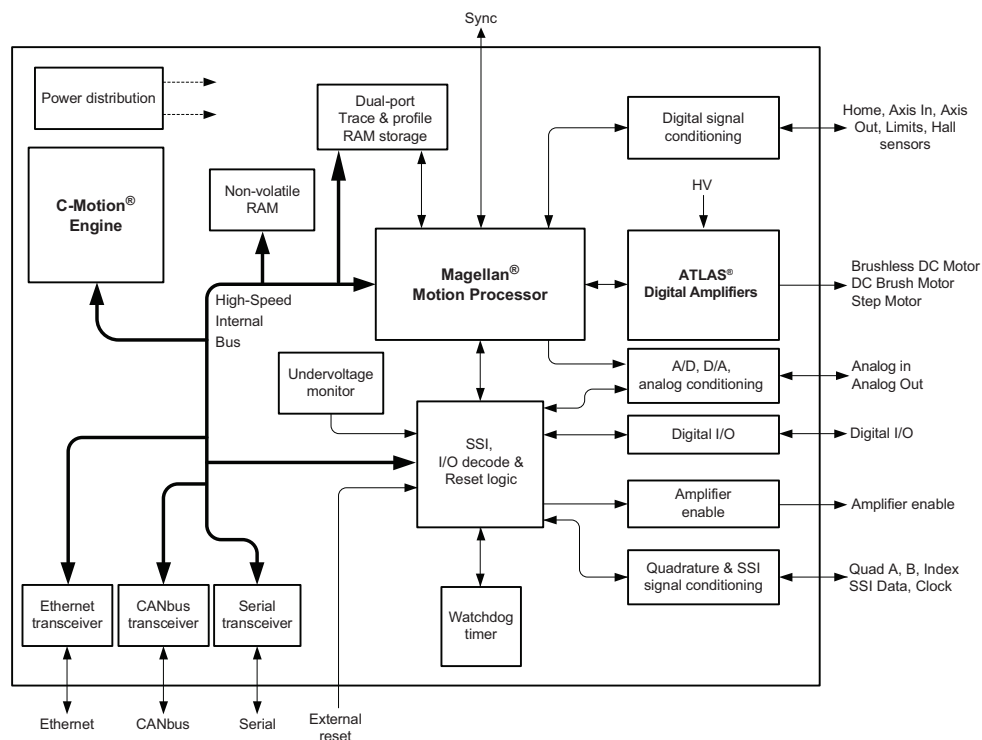
### In This Chapter

- ▶ Board Function Summary
- ▶ Magellan Functions
- ▶ I/O Functions
- ▶ Communications Functions
- ▶ Atlas Amplifier Functions
- ▶ General Board Functions
- ▶ C-Motion Engine Functions
- ▶ Software Libraries

The Prodigy/CME Machine-Controllers are high-performance board based motion controllers for DC brush, brushless DC, and step motors. They are based on Magellan Motion Control ICs, 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-board amplifiers.

Machine controller boards include a powerful C-Motion Engine module (CME) which allows C-language application code to be downloaded and executed directly on the board. 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 board:



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

## 2.1 Board Function Summary

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

**Magellan functions** - These are functions which reside in the Magellan Motion Control IC. 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 Control IC'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 board provides sophisticated communication facilities consisting of two serial ports, a CANbus port, and an Ethernet port.

**Atlas Amplifier functions** - The on-board Atlas amplifiers accept motor output commands from the Magellan Motion Control IC 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 Board functions** - These are general purpose board functions such as a dual-ported RAM, a non-volatile RAM, a board reset function, a board 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 board. It can communicate with various resources on the board including the Magellan Motion Control IC, the board's serial, CANbus, and Ethernet ports, and other on-board resources such as the dual-ported RAM.

### 2.1.1 Board Access Basics

Access to the Prodigy/CME Machine-Controller board 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 board's functions. Various board 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 board, or connected to the board, via the serial, CANbus, or Ethernet connections.

A basic communication to the Prodigy/CME Machine-Controller board 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 board, a return communication is sent by the Prodigy/CME Machine-Controller board 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 board. The **Device** resource indicates functionality that is addressed to the entire board, the **MotionProcessor** resource indicates a Magellan Motion Control IC, 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-board registers.

There are ten different PRP actions including **Command**, which is used to send commands to resources such as the Magellan Motion Control IC, **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-board 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 Programming 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 board architecture so that the user may directly access the board functions of interest to them.



In the subsequent sections of this chapter the C-Motion library functions used to access the available board functions will be provided along with the descriptions of the board 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 this register.
0x210	General purpose digital output write mask	Read & Write	The 12-bit output mask is stored in this register.
0x212	General purpose digital output write value	Read & Write	The 12 output bits are stored in this register.
0x218	Amplifier enable signal write value	Read & Write	The 4 Amplifier enable signal output bits are stored in this register.
0x21A	Amplifier enable signal write mask	Read & Write	The 4 Amplifier enable mask bits are stored in this register.
0x220	Bi-directional I/O direction mask	Read & Write	The 8 bi-directional I/O bits are stored in this register.
0x222	Bi-directional I/O direction value	Read & Write	The 8 bi-directional digital I/O bit directions are set with 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 board functions introduced in the sections below.

## 2.2 Magellan Functions

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

- 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 Control IC interfaces with motion hardware components such as feedback encoders and the on-board 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 Control IC. 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 board the Magellan uses its high-speed parallel-word communications mode to connect to the board'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 board 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 board resources.

In the Magellan User 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 board 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 boards 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 Control IC operates and how it is programmed: the *Magellan Motion Control IC User Guide*, and the *Magellan Motion Control IC Programming 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 Control IC 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 1.7, “Preparing the Board for Installation,”](#) for details.

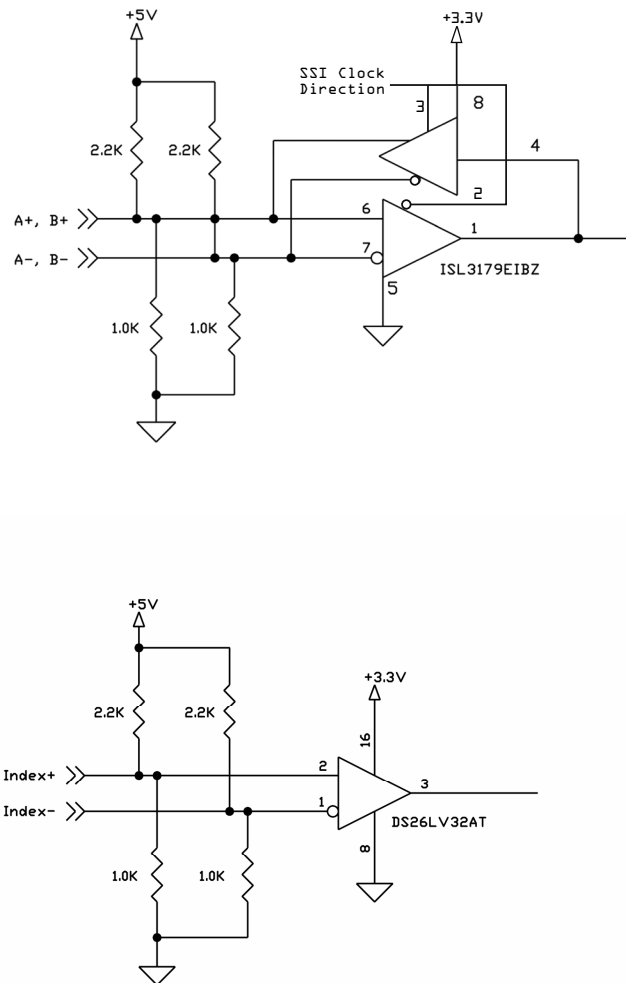
When using the system with differential connections, the polarity of the differential signal can be reversed by swapping the + and - connections. This may be useful for altering the motor and/or encoder direction, however, this same function can also be accomplished through commands to the Prodigy/CME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

See [Chapter 3, Electrical Reference](#), for a complete description of the pinout connections to and from the board.

### 2.2.3.3 Electrical Interfacing

All of the QuadA, QuadB and Index inputs utilize the following on-board 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 board 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-board logic processes the SSI protocol, converting the axis position to a format that allows it to be input to the Magellan Motion Control IC, 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 board:

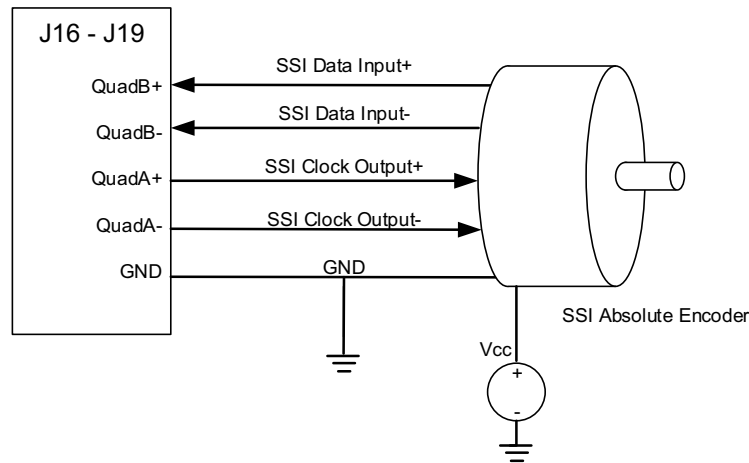
Parameter	Range	Default Value	Comments
SSI clock frequency	500 kHz - 4.0 MHz	500 kHz	The user should set the clock frequency at the highest rate supported by the SSI encoder and cable length they will be using. Check the encoder specification for details.
SSI encoder resolution	8 bits - 31 bits	25 bits	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 boards, the non-clock generating boards 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.

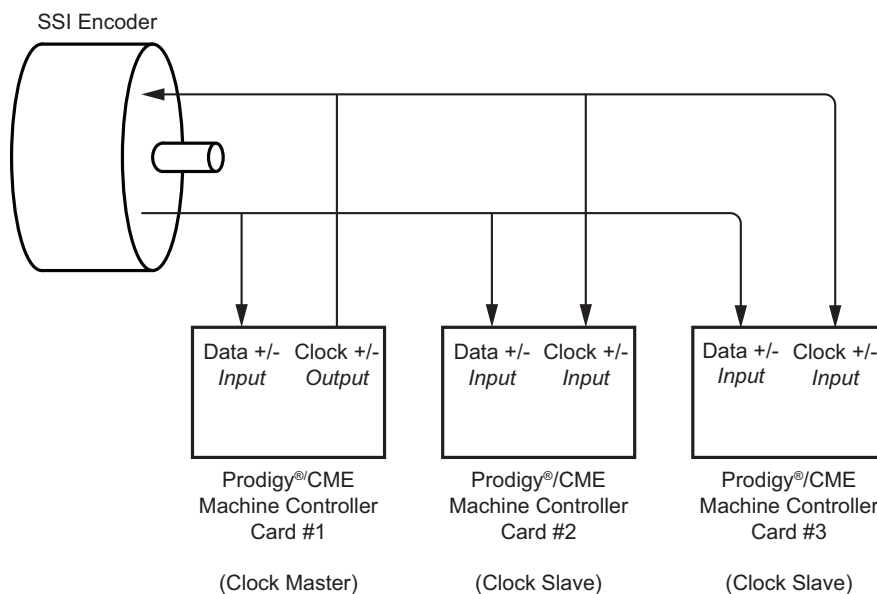


**Figure 2-3:**  
SSI with  
Controller as  
Clock Master

#### 2.2.4.1 Connecting To Multiple Boards

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

This configuration is shown in [Figure 2-4](#).



**Figure 2-4:**  
Connecting  
Multiple Boards

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 Control IC should be set for 32-bit parallel mode to enable proper data transfer from the board'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 Control IC User 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 board. 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 board.

### 2.2.4.5 Electrical Interfacing

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

## 2.2.5 Home, Limits, Hall Sensors

These signals are conditioned by the board and input directly to the Magellan Motion Control IC. The *Magellan Motion Control IC User 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 Control IC User's Guide* for complete information.

### 2.2.5.2 Connections & Associated Signals

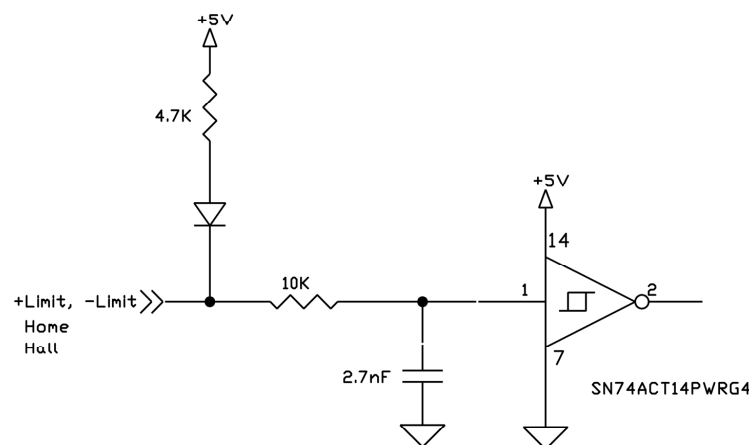
These signals are named *Home I-4*, *PosLim I-4* (positive direction limit input), *NegLim I-4* (negative direction limit input), and *Hall IA-4C* (12 signals in all). They are all located on the Axis Feedback connectors, as are the ground signals that should be used in conjunction with these inputs.

These signals are single-ended digital inputs to the board. 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 board.

### 2.2.5.3 Electrical Interfacing

All of the home, limits, and Hall sensor inputs utilize the following on-board 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 Control IC 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 Control IC 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 Control IC 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 board.

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

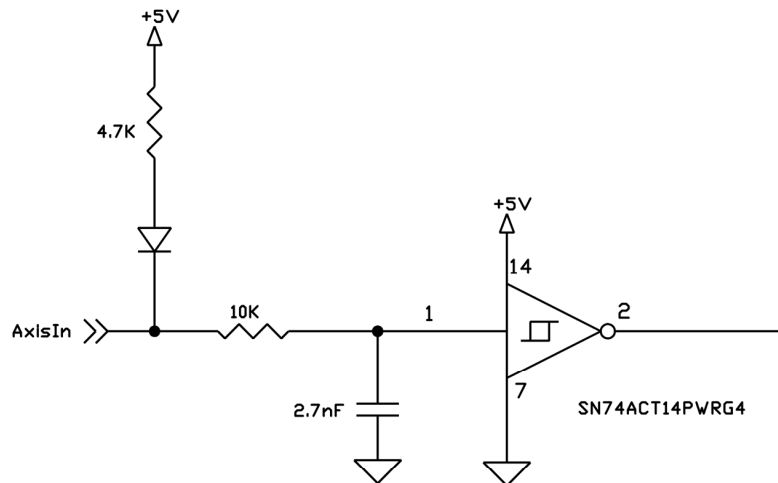
The input signals are pulled up through 4.7k Ohm resistors to 5V. The default power-up value for all *AxisOut* signals is high.

See [Chapter 3, Electrical Reference](#), for a complete description of the pinout connections to and from the board.

### 2.2.6.3 Electrical Interfacing

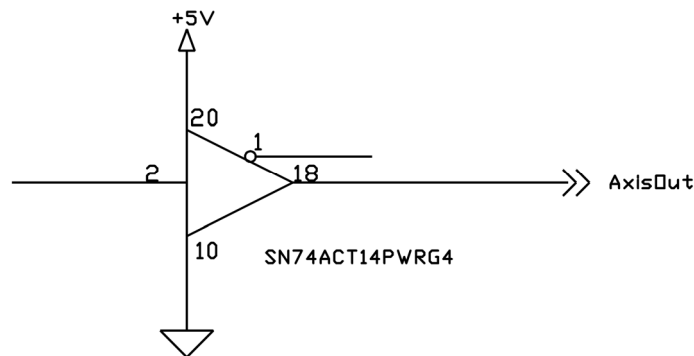
*AxisIn* signal processing is identical to home, limits, and Hall sensor input and is shown in [Figure 2-6](#).

**Figure 2-6:**  
AxisIn Signal  
Schematic



The *AxisOut* signal is generated using the following circuitry: Note that the *AxisOut* signal can source  $\pm 24\text{mA}$ .

**Figure 2-7:**  
AxisOut Signal  
Schematic



## 2.2.7 Atlas Interfacing

The Magellan Motion Control IC 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 Control IC. 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 Control IC 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 Control IC 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 Control IC and Atlas digital amplifiers are internal to the board. 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 Control IC 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 Control IC these analog outputs carry the desired voltage or torque for a given axis, and are designed to interface to an off-board motor amplifier.

Analog motor outputs are useful when on-board 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 Control IC 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 board 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 Control IC as described above. The other source is via the PRP interface. The default is control

by the Magellan Motion Control IC. 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 board 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 Control IC User's Guide* for more information.



If the Magellan Motion Control IC 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

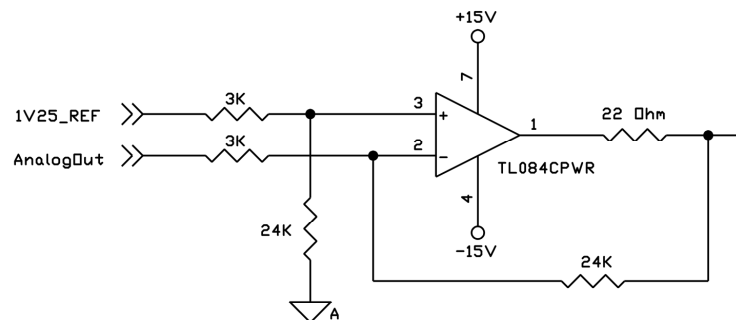
These signals are named AnalogOut1-8+ and AnalogOut1-8, and are located on the Amplifier IO connector. They are differential analog outputs which vary between -10V and +10V.

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 board 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-board 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 board 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 board 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 Control IC explicitly through a user-initiated action. Note that this type of reset is different than a full board reset. See [Section 2.6.5, “Board Reset,”](#) for a description of the board reset function.

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

Signal Name	State
AxisOut I-4	High
AnalogOut I-8	Depends on analog output source
AmpEnable I-4	No change

### 2.2.10.1 C-Motion Commands

The C-Motion command **MPDeviceReset** causes a reset of the Magellan Motion Control IC.

### 2.2.10.2 Connections & Associated Signals

Although as noted in the table above some board 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 board 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 current of 24mA, and an output source current of 5mA..

## 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. Note that when the input is low, the bit value will be 0, and when the input is high (5V), the bit value will be 1.

Signal Name	General IO Connector Pin No.	Bit No.
DigitalIn1	28	0
DigitalIn2	29	1
DigitalIn3	43	2
DigitalIn4	44	3
DigitalIO1	9	8
DigitalIO2	10	9
DigitalIO3	11	10
DigitalIO4	25	11
DigitalIO5	26	12
DigitalIO6	27	13
DigitalIO7	40	14
DigitalIO8	41	15

## 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 the write value and write mask registers described above must be used.

For the 8 bi-directional I/O bits, selecting the direction of these bits is accomplished using a mask/value method similar to the digital output writes described above. The desired direction (input or output) for each bit is written using the I/O direction value register, and the bits that are to be written to in the I/O direction value register are specified via a separate I/O direction mask. Only the mask bits that are 1 will result in the corresponding I/O direction bits being updated by the write command. Note however from the table below that 1 direction value and mask bit controls two output bits. For example setting the direction at bit 8 in the direction register and mask determines the direction of both DigitalIO1 and DigitalIO2.

The following table correlates the signals with the bits of the write value register, the write mask, the I/O direction register and the I/O direction mask.

Signal Name	General IO Connector Pin No.	Bit No. of Write Register and Mask	Bit No. of I/O Direction Register and Mask
DigitalOut1	13	0	N/A
DigitalOut2	14	1	N/A
DigitalOut3	15	2	N/A
DigitalOut4	30	3	N/A
DigitalIO1	9	8	8
DigitalIO2	10	9	8
DigitalIO3	11	10	9
DigitalIO4	25	11	9
DigitalIO5	26	12	10
DigitalIO6	27	13	10
DigitalIO7	40	14	11
DigitalIO8	41	15	11

For the bi-directional signals DigitalIO1-8, each 0 bit value written to the I/O direction register selects the corresponding signal as an input, and a 1 selects it as an output.

For output signals DigitalOut1-4, each 0 bit value written results in a high (5V) output in the corresponding signal and a 1 results in a low (0V) output.

For the bi-directional signals DigitalIO1-8, for each signal selected as an output, each 0 bit value written to the value write register results in a low (0V) output and a 1 results in a high (5V) output. Note that this is the opposite sense of the DigitalOut1-4 signals. Bi-directional signals defined as inputs are not affected by a write to the value write register.

For information on digital I/O value and mask register addressing see [Section 2.1.2, “Peripheral I/O Space.”](#)

### 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.

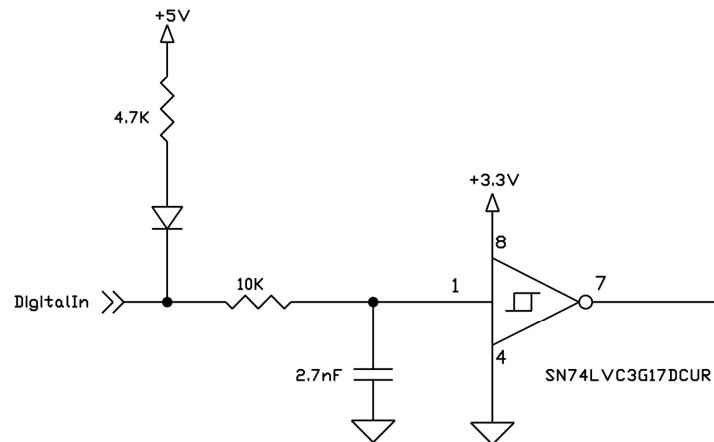
See [Chapter 3, Electrical Reference](#), for a complete description of the pinout connections to and from the board.

### 2.3.3.3 Electrical Interfacing

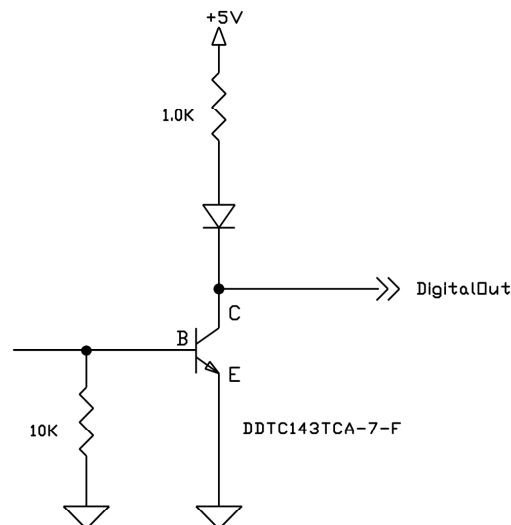
The digital inputs are open-collector logic. When an output pin is set to 1 (high), the output transistor is off and the pin can be used as an input. When an output pin is set to 0 (low), the output transistor is on which pulls the signal to ground and the pin cannot be used as an input.

The digital signals are processed using the following circuitry:

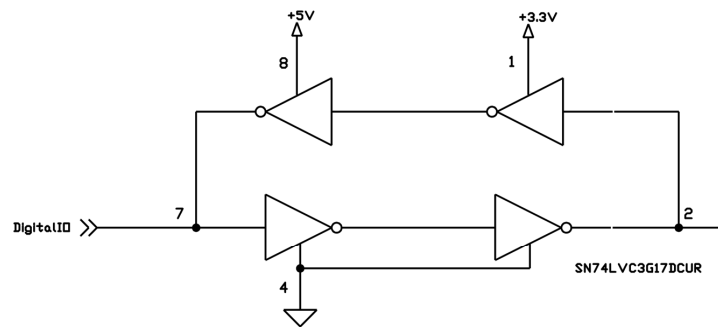
**Figure 2-9:  
DigitalIn(1-4)  
Interface  
Schematic**



**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 board 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 board. 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

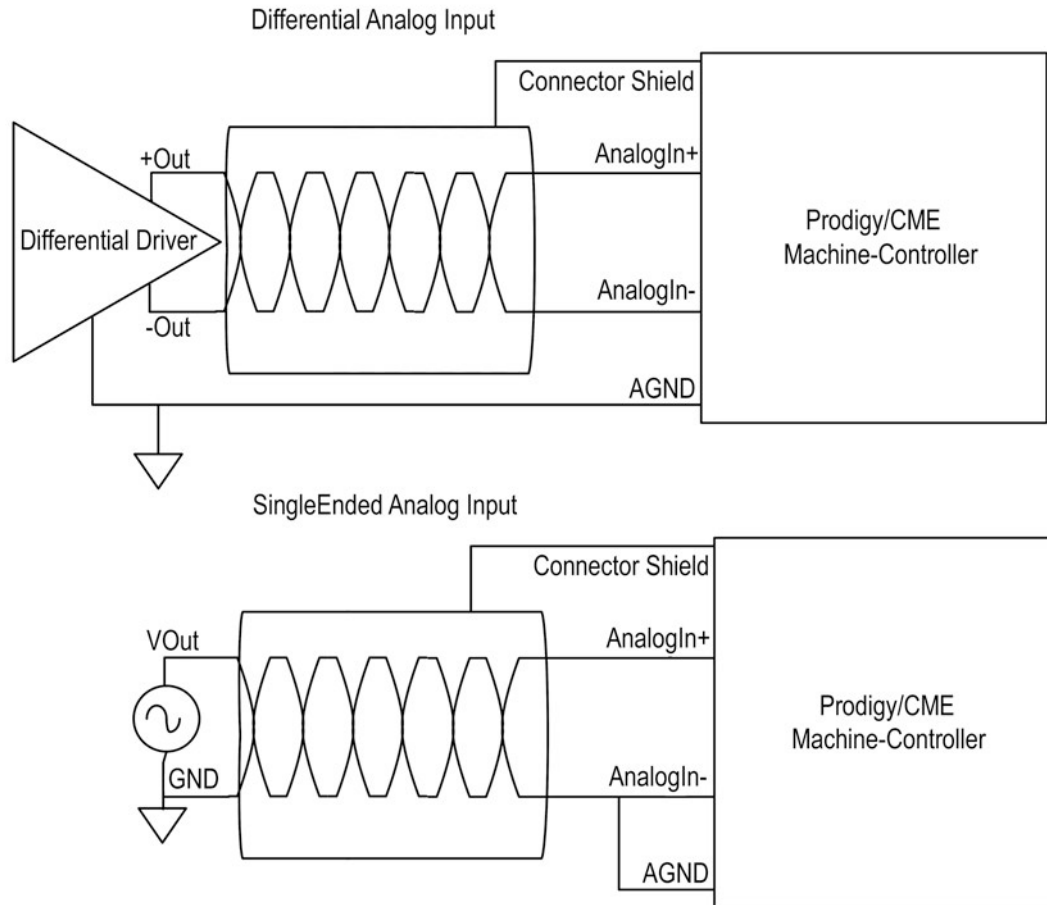
To determine the numerical value that will be read by the Prodigy/CME Machine-Controller board given a specific voltage at the input pins, the following formula is used:

$$\text{ReadValue} = \text{AnalogVoltage} * 32,767 / 10.0\text{V}$$

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

$$\text{AnalogVoltage} = \text{ReadValue} * 10.0\text{V} / 32,767$$

**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**.

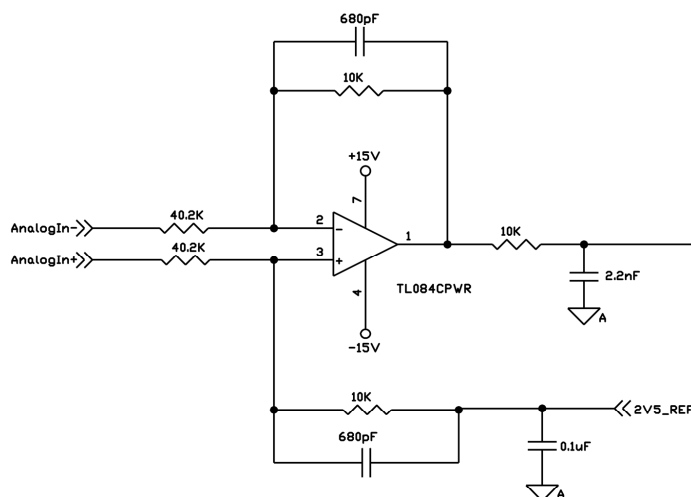
Using the returned *Peripheral* handle, the user can read one or more 16-bit word values representing the current analog voltage using the **PeriphRead** command.

#### 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 board.

### 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 Control IC 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 Control IC.

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 board given a specific commanded output value the following formula is used:

$$\text{AnalogVoltage} = \text{WriteValue} * 10.0\text{V} / 65,535$$

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

$$\text{WriteValue} = \text{AnalogVoltage} * 65,535 / 10.0\text{V}$$

### 2.3.5.1 AnalogOut Enable

The Prodigy/CME Machine-Controller boards 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 board reset, or via the external Reset signal. See [Section 2.6.5, “Board 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`.

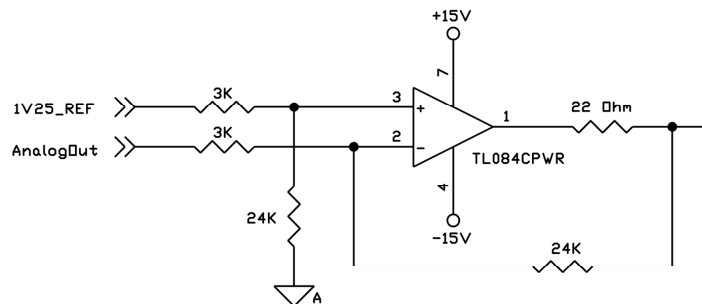
### 2.3.5.3 Connections & Associated Signals

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 board.

### 2.3.5.4 Electrical Interfacing

**Figure 2-14:**  
Analog Output  
Interface



## 2.4 Communications Functions

The Prodigy/CME Machine-Controller board 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 boards 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 board 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 board.

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 board retrieves default information for the serial ports from the on-board non-volatile RAM. To simplify startup these default values can be changed by the user. See [Section 2.6.6, “Setting Board Defaults,”](#) for information.

### 2.4.1.1 C-Motion Commands

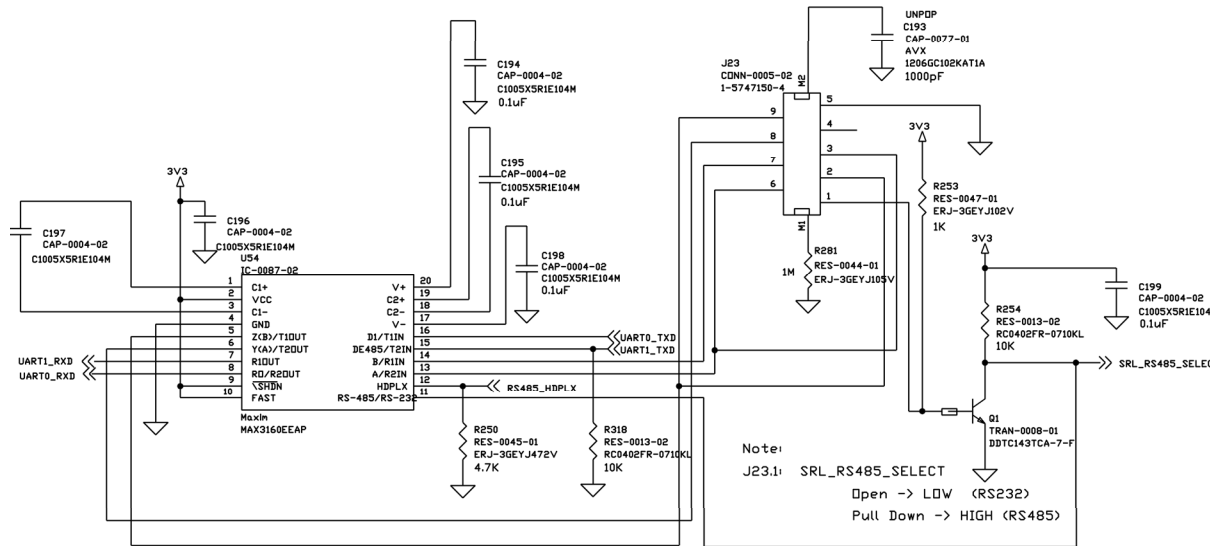
To create a serial port peripheral connection the above parameters are specified in the C-Motion command **PeriphOpenCom**. Messages to and from the Serial port are transmitted via the **PeriphSend** and **PeriphReceive** commands. Whenever a new serial port peripheral is opened its previous function 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 Programming 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.

### Figure 2-15: Serial Interface Schematic



## 2.4.2 CANbus Communications

The Prodigy/CME Machine-Controller board 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 board retrieves default information for the CANbus port from the board's non-volatile RAM. See [Section 2.6.6, “Setting Board Defaults,”](#) for more information on changing these default values.

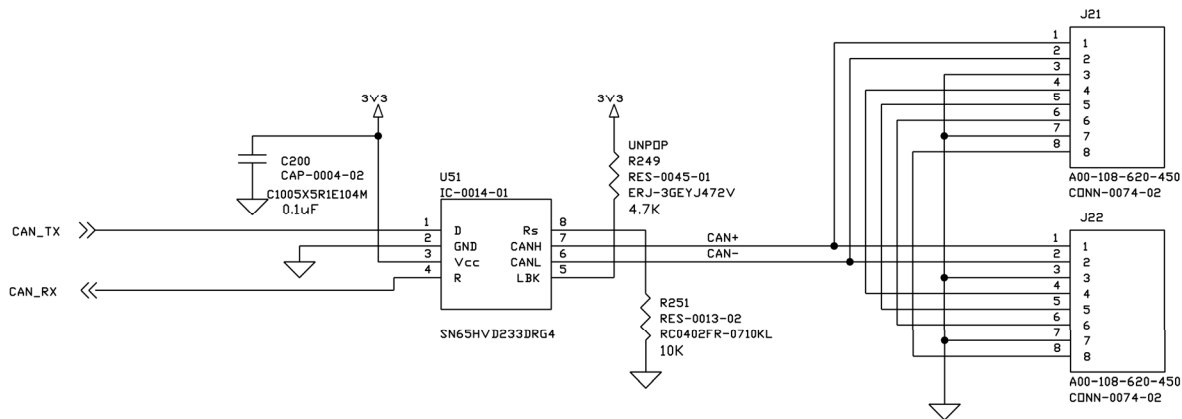
### 2.4.2.1 C-Motion Commands

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 boards support two different Ethernet protocols, TCP (Transmission Control Protocol) and UDP (User Datagram Protocol). TCP is typically used for primary Ethernet communications to the board, 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 board, 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 board:

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 board retrieves default information for the Prodigy/CME Machine-Controller Ethernet port from the board's non-volatile RAM. To change these default values see [Section 2.6.6, “Setting Board Defaults.”](#)

### 2.4.3.1 TCP Connection Keep-Alive

All prodigy/CME board 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 board 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 boards 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 board'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 Board 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 Programming 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-board 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<sup>2</sup>t 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 control ICs. 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.

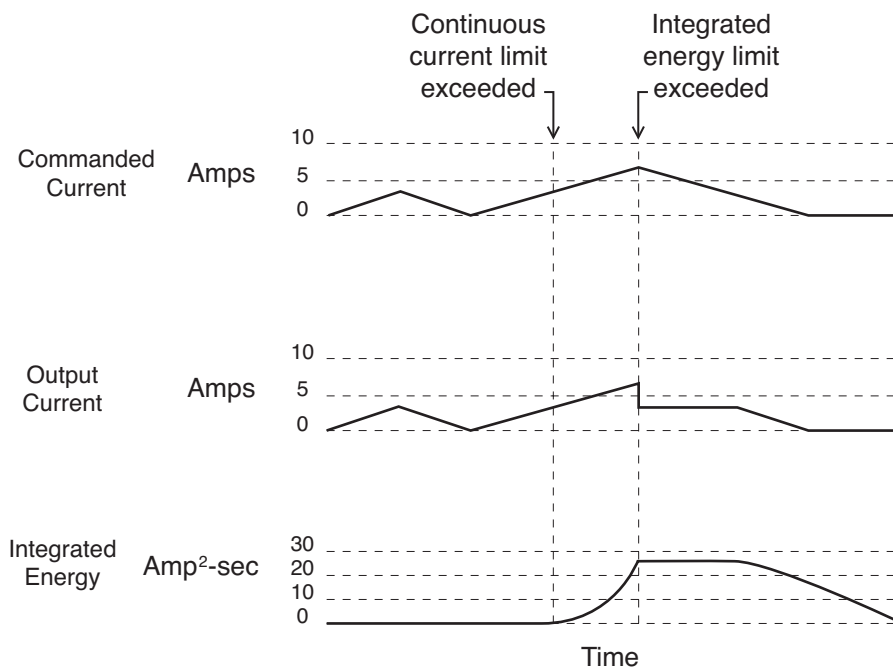
All other aspects of this feature are the same as for overvoltage sense. Just as for overvoltage conditions, it is the user's responsibility to determine the seriousness of, and appropriate response to, an undervoltage condition.

### 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 <sup>2</sup> Sec	139 Amp <sup>2</sup> Sec
DC Brush	12.0 Amps	14.0 Amps	144 Amp <sup>2</sup> Sec	144 Amp <sup>2</sup> Sec
Step Motor	8.0 Amps	9.0 Amps	149 Amp <sup>2</sup> Sec	149 Amp <sup>2</sup> Sec

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.



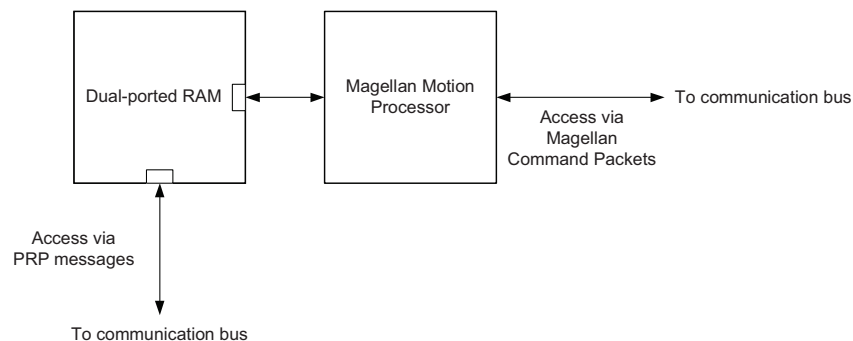
## 2.6 General Board Functions

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

### 2.6.1 Dual-Ported RAM (DPRAM)

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

**Figure 2-18:**  
On-board Dual-ported Memory



The dual-ported RAM is most often used to allow various Magellan Motion Control IC 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-board host using the Prodigy/CME Machine-Controller board'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 Control IC, see the "Trace Capture" section of the *Magellan Motion Control IC User 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 board 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 board.

#### 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 Control IC User Guide* for more information on these commands. In this read



configuration, the Magellan Motion Control IC stores data to the DPRAM autonomously, and the host controller reads the data using the Magellan Motion Control IC as well.

An alternate path for reading from, or writing to, the dual ported RAM is via the Prodigy/CME Machine-Controller board's high speed communication bus. In this mode the dual-port 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 Control IC User 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 Programming 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 boards have a general purpose 4,094 byte memory that retains its contents after a board power down or reset. This memory is useful for storing parameters that are set only occasionally and stay with the board, 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 Programming 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 boards 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**.

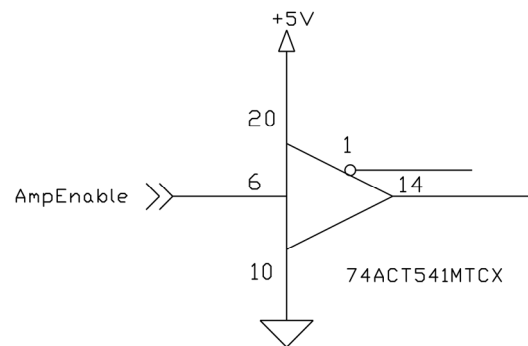
### 2.6.3.2 Connections & Associated Signals

*AmpEnable1-4* are direct digital outputs and appear on the board'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 board.

### 2.6.3.3 Electrical Interfacing

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



**Figure 2-19:**  
AmpEnable  
Output  
Interfacing

## 2.6.4 Reset Monitor

During normal operations, the Prodigy/CME Machine-Controller board is only reset during power-up. There are, however, several other ways that the Prodigy/CME Machine-Controller board can be reset, many of which are not related to a power cycle. See [Section 2.6.5, “Board Reset,”](#) for a description of how the board 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 board to a known and consistent state.

To determine the cause of a board 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	0-9	Reserved
	10	User code exception
	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 board-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 board watchdog timeout. See <a href="#">Section 2.2.9, “Magellan Watchdog Timer,”</a> for a description.

Note that after a board 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 board powerup, a check of this register by the user application code will allow anomalous occurrences of a board 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 Board Reset

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

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

Signal Name	State
AxisOutI-4	High
AnalogOutI-8	0.0 volts
DigitalOutI-4	Low
AmpEnableI-4	High

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

In addition, upon a board reset all board default parameters are reloaded. See [Section 2.6.6, "Setting Board Defaults,"](#) for more information on default values.

#### 2.6.5.1 C-Motion Commands

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

#### 2.6.5.2 Connections & Associated Signals

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

### 2.6.6 Setting Board Defaults

There are a number of user-settable parameters that are saved by the board in non-volatile RAM and that are utilized after a powerup, or after a board 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

Parameter	Factory Default Value
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
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 board reset.

The default values are stored in the board'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 Programming Reference*.

## 2.6.7 Board Undervoltage Monitor

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

If a board undervoltage condition occurs the board will be automatically reset. See [Section 2.6.5, “Board Reset,”](#) for more information on the signal and other conditions affected by a board 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 board and determine the cause of the fault before repowering the board.



### 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 Board 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 board 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 board resource information that may be queried is the board logic version, the device firmware version, and the dual-ported RAM size.

### 2.6.8.1 C-Motion Commands

To query the board 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

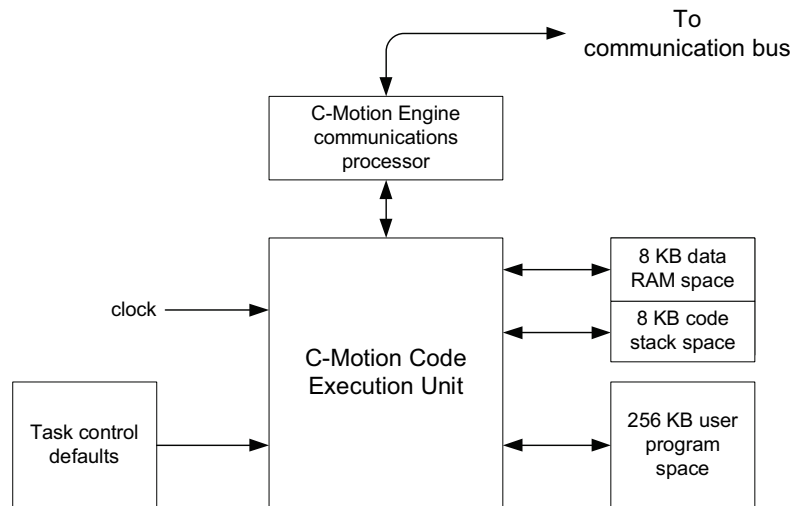
The C-Motion Engine on the Prodigy/CME Machine-Controller board allows C-Motion code to be downloaded and executed on the board. The C-Motion Engine is a powerful and flexible engine that can be used to:

- Operate Prodigy motion boards in a standalone mode
- Offload time-critical code from the host to the motion board
- 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 Control IC with higher level functions such as contouring, macros, or other complex behaviors
- Lower system cost by combining a motherboard function with a dedicated motion board function in a single-board 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 Control IC, and access other on-board resources such as the dual-ported RAM.

Creating, compiling, downloading, and verifying a specific user C-Motion application on a Prodigy/CME Machine-Controller board 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 board. [Figure 2-20](#) provides an overview of the architecture of the C-Motion Engine.



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

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 board 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 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 board. This C-Motion Engine-initiated reset is identical to the reset that occurs after sending a board 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, “Board 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 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 board default parameters, so to change these parameters the C-Motion command **DeviceSetDefault** is used. See [Section 2.6.6, “Setting Board 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 Programming 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 Control IC 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 board’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 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 boards 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 Programming Reference*.

### 2.7.8 C-Motion Engine Heartbeat LED

The Prodigy/CME Machine-Controller board 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 boards. In addition, a special library of commands is used to access Magellan Motion Control IC functions called VB-Motion, and C-Motion.

The *PMD Resource Access Protocol Programming Reference*, in addition to providing detailed format and function descriptions of the PRP messages associated with the Prodigy/CME Machine-Controller boards, 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 board 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 Control IC, 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 board. 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

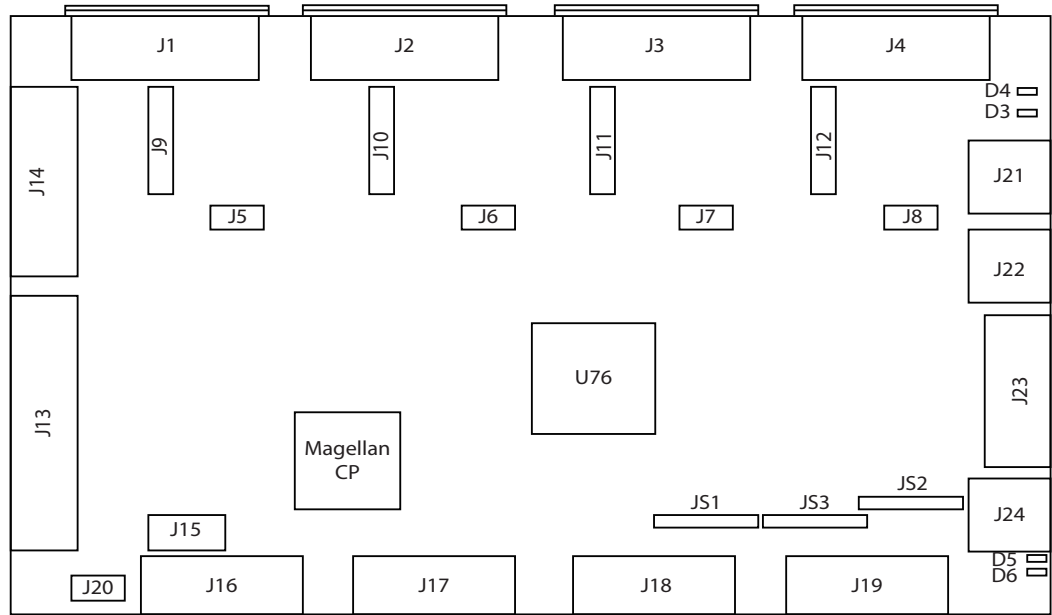
## In This Chapter

- ▶ User-Settable Components
- ▶ Connectors
- ▶ Motor Connections
- ▶ Cables
- ▶ Absolute Maximum Ratings
- ▶ Environmental and Electrical Ratings
- ▶ Certifications & Compliance
- ▶ Mechanical Dimensions
- ▶ L-Bracket Mechanical Dimensions
- ▶ Atlas Mechanical Dimensions

## 3.1 User-Settable Components

Figure 3-1 illustrates the locations of the principal components of the Prodigy/CME Machine-Controller boards. The user-settable components of the board 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 Board

### 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.

#### 3.1.1.1 Differential Encoder Connections

Differential encoder connections are detailed in the following table.

Signal	J16 - J19			
	Axis 1	Axis 2	Axis 3	Axis 4
QuadAn+	J16-1	J17-1	J18-1	J19-1
QuadAn-	J16-2	J17-2	J18-2	J19-2
QuadBn+	J16-3	J17-3	J18-3	J19-3
QuadBn-	J16-4	J17-4	J18-4	J19-4
Indexn+	J16-6	J17-6	J18-6	J19-6
Indexn-	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

#### 3.1.1.2 Single-Ended Encoder Connections

Single-ended encoder connections are detailed in the following table.

Encoder connections when using single-ended encoder input:

Signal	J16 - J19			
	Axis 1	Axis 2	Axis 3	Axis 4
QuadAn	J16-1	J17-1	J18-1	J19-1
QuadBn	J16-3	J17-3	J18-3	J19-3
Indexn	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

#### 3.1.1.3 Using Both Single-Ended and Differential Encoder Connections

When both single-ended and differential encoders are used on the same board a special arrangement of the connections is needed. If Axis 1 or Axis 2 has a single-ended encoder, remove the resistor packs installed on JS1 and JS3. JS1, JS2, and JS3 are the connectors that receive the RS1, RS2, and RS3 resistor packs, respectively. If Axis3 or Axis4 has a single-ended encoder, remove the packs installed on JS2 and JS3.

For any axis that has a differential encoder and the corresponding resistor pack has been removed, a 120 ohm resistor needs to be installed as follows:

	Axis 1	Axis 2	Axis 3	Axis 4
Channel A	JS1-1,2	JS1-5,6	JS2-1,2	JS2-5,6
Channel B	JS1-3,4	JS1-7,8	JS2-3,4	JS2-7,8
Index (Z)	JS3-1,2	JS3-3,4	JS3-5,6	JS3-7,8

Every cell in the table above represents an installed 120 ohm resistor at that location. For example JS1-1,2 implies that one terminal of the resistor is connected to JS1-1 and the other terminal is connected to JS1-2.

## 3.2 Connectors

There are 24 user-accessible connectors on the Prodigy/CME Machine-Controller board. See [Figure 3-1](#) for the specific locations of the connectors on the board. 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 board plus Axis 1 Atlas unit. It accepts DC supply in the range of +12 – 56 VDC.
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 $\pm 10V$ 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 Axis 1.
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.

Connector Name	Connector #	Functionality
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 HVI 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
<b>J1</b>		
1	GND	Power return for HVI, Motor A, Motor B, Motor C and Motor D.
2	GND	Power return for HVI, Motor A, Motor B, Motor C and Motor D.
3	HVI	DC power to axis 1 Atlas module, referenced to GND.
4	HVI	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	~SPICSI	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
<b>J2</b>		
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.

Pin	Connection	Description
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
<b>J3</b>		
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
<b>J4</b>		
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 board, 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
<b>J5</b>		
1	HV1	Provides DC power to the board and Axis 1 Atlas module
2	GND	Ground

#### 3.2.2.2 Axis 2 Power Connector

Pin	Connection	Description
<b>J6</b>		
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
<b>J7</b>		
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
<b>J8</b>		
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
<b>J16</b>		
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 Index- 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
<b>J17</b>		
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
<b>J18</b>		
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	NegLim3	Axis 3 Negative direction limit switch input
14	Vcc	+5V
15	GND	Ground

### 3.2.3.4 Axis 4 Feedback Connector

Pin	Connection	Description
<b>J19</b>		
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		
<b>J9</b>		
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		
<b>J10</b>		
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		
<b>J11</b>		
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		
<b>J12</b>		
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
<b>J13</b>					
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-board synchronization and an external RESET signal. Depending on whether the board 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 boards 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 boards 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
<b>J23</b>				
1	RS485Select	Open (default) selects RS232 mode for both Serial1 and Serial2	Tie to ground selects RS485 mode	Tie to ground selects RS485 mode
2	Sr1IXmt	Serial 1 transmit output	no connect	no connect
3	Sr1IRcv	Serial 1 receive input	no connect	no connect
4	No connect	no connect	no connect	no connect
5	GND	Ground	Ground	Ground
6	RS485Rcv <sup>+</sup>	no connect	Positive (non-inverting) receive input	no connect
7	Sr12Rcv/RS485Rcv <sup>-</sup>	Serial 2 receive input	Negative (inverting) receive input	no connect
8	Sr12Xmt/RS485Xmt <sup>+</sup>	Serial 2 transmit output	Positive (non-inverting) transmit output	Positive transmit/receive
9	RS485Xmt <sup>-</sup>	no connect	Negative (inverting) transmit output	Negative transmit/receive

### 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 board 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
<b>J21, J22</b>		
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
<b>J24</b>		
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-I10-01-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-	J1-2	J2-2	J3-2	J4-2
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
Phase A	J14-7	J14-8	J14-9	J14-15
Phase B	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 board'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 board's CANbus connector and has RJ45 connectors on both ends.
Cable-RJ45-02-R	Ethernet connector. This cable connects to the board's Ethernet connector.
Cable-I007-01	Serial connector. This cable connects to the board's serial connector.

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

### 3.4.1 Cable-5001-01

<b>PMD Part #:</b> Cable-5001-01	<b>J5-J8</b>	<b>Signal</b>
<b>Description:</b> 2-signal HV Power supply stub cable.	1	HV
<b>Length:</b> 1.0 meter	2	Ground

### 3.4.2 Cable-5002-01

<b>PMD Part #:</b> Cable-5002-01	<b>J9-J12</b>	<b>Signal</b>	<b>Wire Color</b>
<b>Description:</b> This stub cable connects to the Motor Drive Connectors.	<b>Pin</b>		
<b>Length:</b> 1.0 meter	1	Motor A+	Red
<b>Cable:</b> 5 conductor, 4.20mm pitch, 16AWG stranded cables	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



J13 Pin	Signal	J13 Pin	Signal
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	SrlIXmt	2
3	SrlIRcv	3
4	No connect	-
5	GND	5
6	RS485Rcv+	6
7	Srl2Rcv/ RS485Rcv-	7
8	Srl2Xmt/RS485Xmt+	8
9	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, $\pm 3\text{mA}$ 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: $\pm 50\text{mA}$
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)

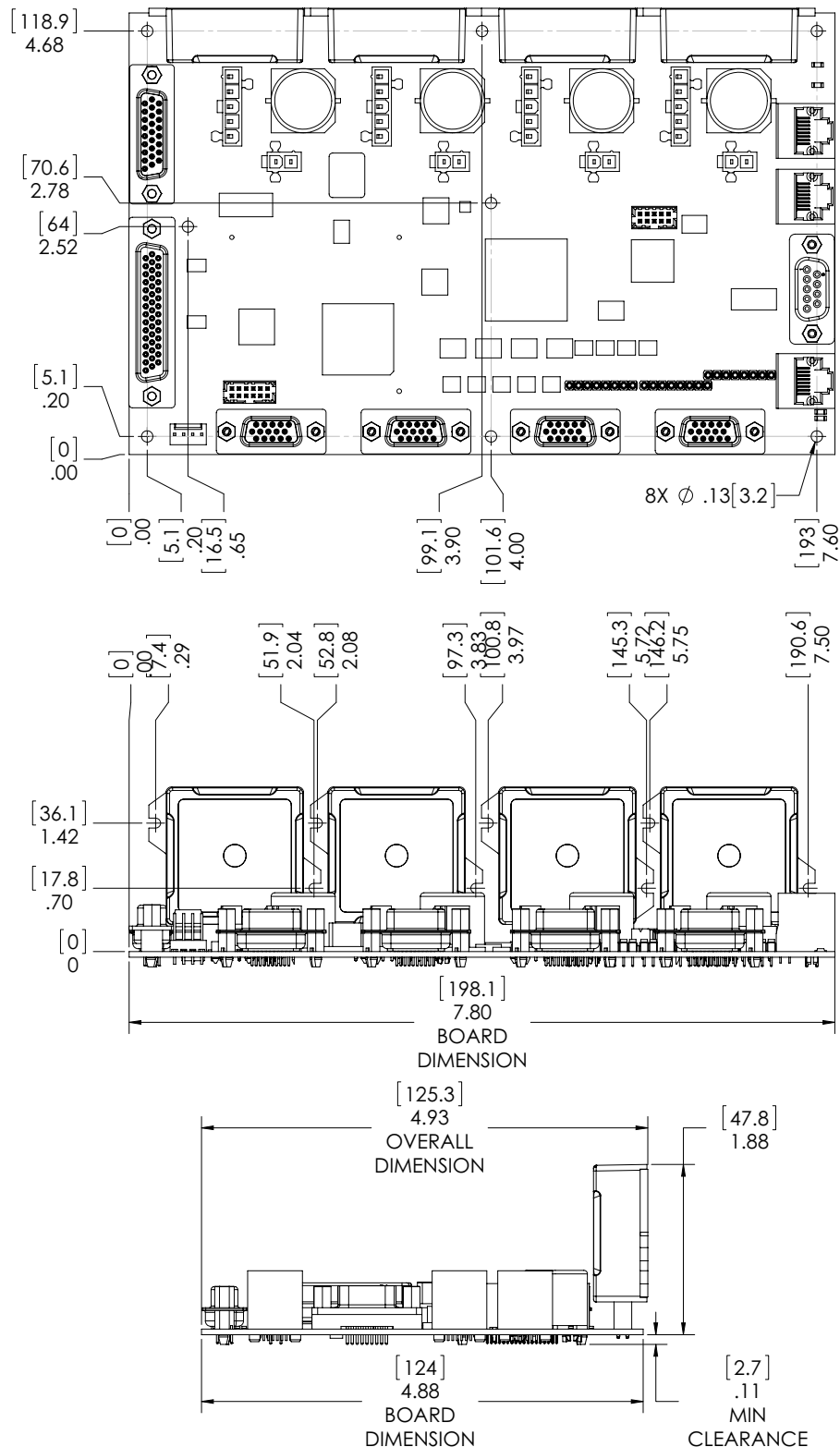
## 3.7 Certifications & Compliance

Specification	Standard
CE	LVD: EN60204-1 EMC-D: EN6100-6-1, EN61000-6-3, EN55011

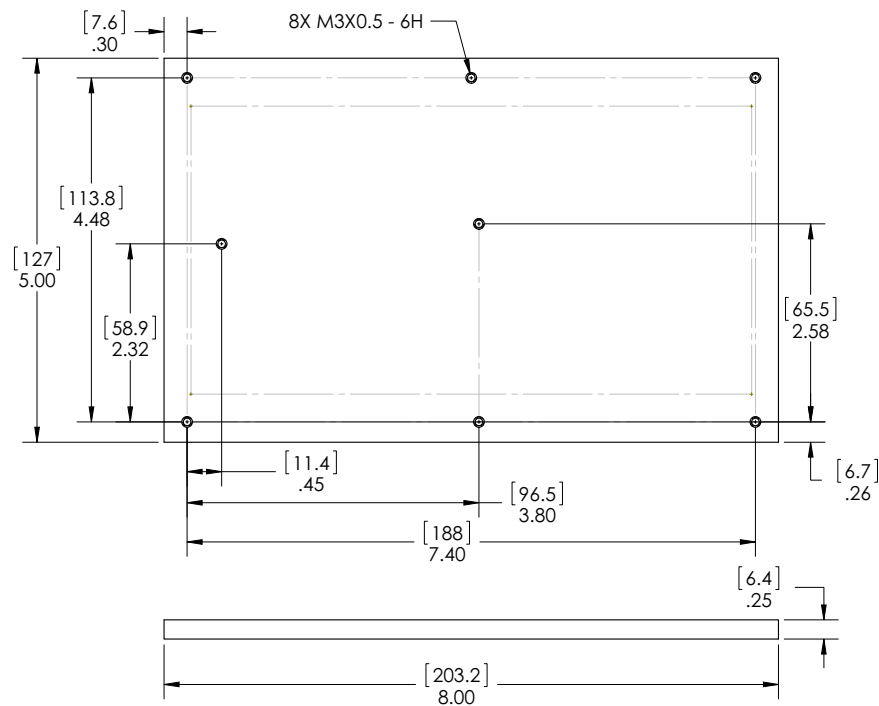
## 3.8 Mechanical Dimensions

The size of the board is 7.80" L x 4.88" W. For purposes of illustration the compact Atlas unit size is shown in the drawings below. Vertical dimensions will be different for installed ultra-compact units mounted on the compact to ultra-compact converter.

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

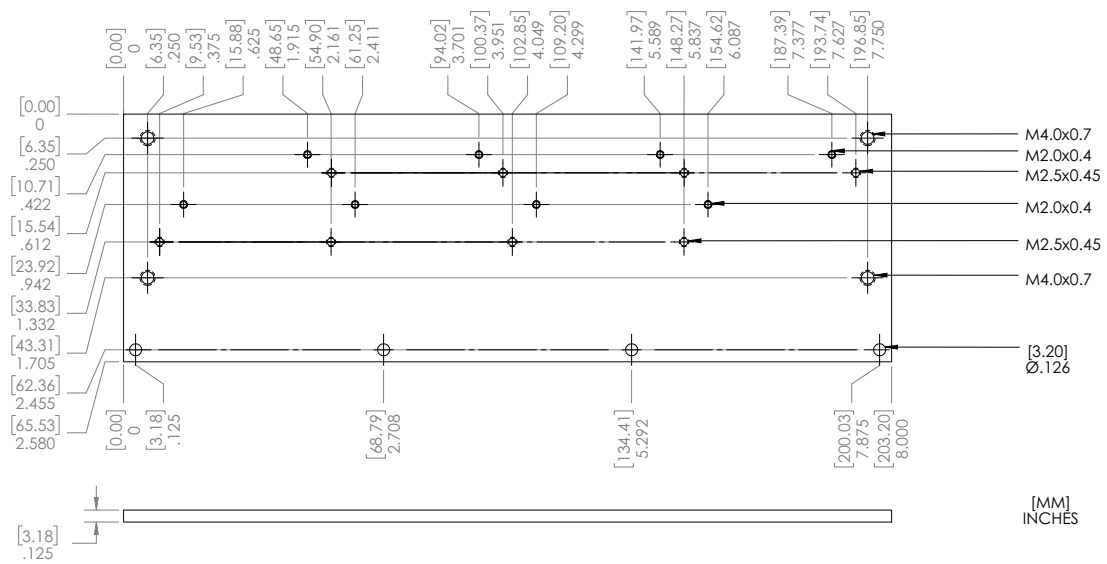


### 3.9 L-Bracket Mechanical Dimensions



**Figure 3-3:**  
**L-Bracket Base**  
**Mechanical**  
**Dimensions**

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



# Index

---

## A

- absolute maximum ratings 90
- analog input 53
- analog output 47, 55
- Atlas amplifier
  - connectors 77, 78
  - electrical installation 15
  - functions 36, 60
  - interfacing 46
  - overview 60
  - sockets 78–79

## B

- brushless DC motors 30, 47
  - Hall sensors 45

## C

- card
  - access 36
  - function summary 36
  - reset 68
  - resource revision information 69
  - setting defaults 68
  - types 9
- C-Motion Engine
  - functions 70
  - hardware configuration 70
  - heartbeat LED 74
  - powerup and operation 71
  - task control 72
- commands
  - C-Motion 39
  - Magellan 39
- communication 56
  - CANbus 58–59
  - Ethernet 59–60
  - serial 57–58
- commutation 30, 47
  - brushless DC motors 30
- components table, front of board 19
- connections
  - communications 24

- motor drive 22
- motor feedback 21
- motor power 23
- connectors 77
  - amplifier I/O 84
  - CAN 85
  - Ethernet 86
  - expansion 84
  - feedback 81
  - general I/O 83
  - motor drive 82
  - parts reference 86
  - power 80
  - serial 85
- current foldback 63
  - processing example 63

## D

- debug console window 73
- drive
  - enable 63
  - fault status register 63
  - ratings 61

## E

- electrical ratings 91
- encoder
  - settings 76
- environmental ratings 91

## F

- fault
  - overcurrent 61
  - overtemperature 61
- FaultOut Signal 63

## H

- Hall sensors 45
- hardware accessory products 11
- hardware recommendations 13

## I

- I/O functions 49–56

analog input 53–55  
analog output 55–56  
general-purpose digital 49–52  
internal block diagram 61

## **L**

L-bracket 11  
card assembly 14  
installation 14  
mechanical dimensions 93

## **M**

Magellan  
Atlas interfacing 46  
AxisIn, AxisOut signals 45  
functions 36, 38  
instructions 38  
reset 49  
watchdog timer 48  
memory  
dual-ported 64  
non-volatile 65  
message URL <https://www.pmdcorp.com/company/patents> iii  
motor  
current 63  
motor connection 20  
brushless DC 20  
DC brush 21  
step 21

## **N**

non-volatile initialization storage 64

## **O**

overcurrent fault 61  
overtemperature fault 61  
overvoltage fault 62

## **P**

Peripheral I/O space 37  
pinouts 85  
CAN connectors 85  
Ethernet connector 86  
power  
applying 25  
connections 23  
power-up 64

## **Q**

quadrature encoder  
C-Motion commands 40  
connections 40  
signals 40–41

## **R**

ratings  
drive 61  
resistor 20  
resistor packs 75  
settings 20, 76

## **S**

safety processing functions 61  
current foldback 63  
drive  
enable 63  
fault status register 63  
FaultOut Signal 63  
overcurrent fault 61  
overtemperature fault 61  
overvoltage fault 62  
undervoltage fault 62  
watchdog timeout 63  
software  
libraries 74  
SSI 41–44

## **T**

TCP 60

## **U**

undervoltage fault 62  
user application code 72, 73  
developing 31–33  
user-settable components 75  
user-settable parameters 68

## **W**

watchdog  
timeout 63  
timer 48