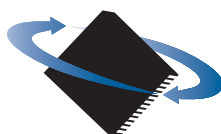


# **Prodigy<sup>®</sup>/CME Stand-Alone Motion Board User Guide**



**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 PMD.

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.

Prodigy, Magellan, ION, Magellan/ION, 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<sup>®</sup> Motion Control IC User Guide**

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

### **Magellan<sup>®</sup> 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/CME product commands, with software architecture overview, command syntax, and examples.

### **C-Motion<sup>®</sup> 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 Stand-Alone Motion Board Overview .....	9
1.2 Prodigy/CME Stand-Alone Developer Kit .....	10
1.3 Accessory Products .....	10
1.4 Installation Overview .....	11
1.5 Recommended Hardware .....	11
1.6 Software Installation .....	12
1.7 Preparing the Board for Installation .....	14
1.8 Connection Summary .....	16
1.9 Applying Power .....	19
1.10 First-Time System Verification .....	20
1.11 Developing User Application Code .....	26
<b>Chapter 2. Operation</b> .....	<b>29</b>
2.1 Board Function Summary .....	30
2.2 Magellan Motion Control IC Functions .....	32
2.3 Magellan-Controlled Functions .....	33
2.4 C-Motion Engine Functions .....	39
2.5 Communications Functions .....	43
2.6 General Board Functions .....	46
2.7 Signal Processing and Hardware Functions .....	49
2.8 Software Libraries .....	51
<b>Chapter 3. Accessing Board Resources</b> .....	<b>53</b>
3.1 Resource Addressing .....	53
3.2 Accessing the Communications Ports .....	55
3.3 Accessing On-Board Resources .....	58
3.4 Accessing Magellan-Attached Devices .....	59
3.5 PRP Communication Formats .....	61
<b>Chapter 4. Electrical Reference</b> .....	<b>65</b>
4.1 User-Settable Components .....	65
4.2 Connectors .....	68
4.3 Connections Summary—Motor Amplifiers .....	76
4.4 Cables .....	79
4.5 Environmental and Electrical Ratings .....	84
4.6 Certifications & Compliance .....	84
4.7 Mechanical Dimensions .....	85
4.8 User I/O Memory Map .....	85
<b>Chapter 5. Interconnect Module</b> .....	<b>87</b>
5.1 IM-1000 Interconnect Module .....	87
5.2 IM-600 Interconnect Module .....	89
<b>Index</b> .....	<b>91</b>




---

This page intentionally left blank.

# List of Figures

---

1-1	Prodigy/CME Stand-Alone Board Component Location .....	14
1-2	Switch settings .....	16
1-3	Serial Port Connection .....	18
1-4	Serial and Ethernet Connection .....	19
1-5	Two Ways to Locate the Code on the Prodigy/CME Stand-Alone Board .....	27
2-1	Prodigy/CME Stand-Alone Board Internal Block Diagram .....	30
2-2	On-board Dual-ported Memory .....	38
2-3	Overview of C-Motion Engine Architecture .....	40
3-1	Outgoing and Returning PRP header formats .....	54
3-2	Example Network Configuration .....	57
3-3	Host Controller & On-board Resources .....	58
3-4	Host Controller & Magellan-attached Devices .....	60
3-5	PRP Header over Serial Format .....	61
4-1	Components and layout, front of board .....	65
4-2	Switch Settings by Output Type .....	68
4-3	Sync I/O connector to three boards .....	74
4-4	Prodigy/CME mechanical dimensions .....	85
5-1	IM-1000 location of components .....	87
5-2	IM-600 location of components .....	89



---

This page intentionally left blank.



# 1. Installation

1

## In This Chapter

- ▶ Prodigy/CME Stand-Alone Board Types
- ▶ Prodigy/CME Stand-Alone Developer Kit
- ▶ Accessory Products
- ▶ Installation Overview
- ▶ Recommended Hardware
- ▶ Software Installation
- ▶ Preparing the Board for Installation
- ▶ Connection Summary
- ▶ Applying Power
- ▶ First-Time System Verification
- ▶ Developing User Application Code

## 1.1 Prodigy/CME Stand-Alone Motion Board Overview

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

Prodigy/CME Stand-Alone boards come in various configurations of axis number (from 1 to 4). The following table provides information on the specific board versions that are available. It shows the relationship between board part numbers, Magellan Motion Control IC part numbers contained on the board, the number of axes supported, and the type of motors supported. In the motor type column, “all motor types” means DC brush, brushless DC, microstepping, and step (pulse & direction) motors.

Prodigy/CME Stand-Alone Board P/N	Magellan P/N	Number of Axes	Motor Type
PR13V58120	MC58120	1	All motor types
PR13V58220	MC58220	2	All motor types
PR13V58320	MC58320	3	All motor types
PR13V58420	MC58420	4	All motor types

### 1.1.1 Supported Motor Types

The Prodigy/CME Stand-Alone boards support four different motor types; DC Brush, Brushless DC, step motors with a microstepping output, and step motors with a pulse and direction output. Below is a summary of each of these four motor types

**DC Brush:** output is a single-phase motor command, either in PWM (pulse width modulated), or analog ( $\pm 10V$ ) output format. The output is intended to control DC Brush motors, or brushless DC motors with an amplifier that performs commutation.

**Brushless DC:** output is multi-phase motor command signals, either in PWM (pulse width modulated), or analog ( $\pm 10V$ ) output format, using Hall-based or sinusoidal commutation. The output is intended to interface with brushless DC amplifiers and motors.

**Microstepping:** output is multi-phase analog ( $\pm 10V$ ) or PWM (pulse width modulation) waveforms. The output is intended to control 2- or 3-phase step motors using amplifiers which accept this command format.

**Pulse & direction:** output is standard pulse & direction signals intended to interface with amplifiers which accept this command format.

For complete information on motor output formats and other information, see the *Magellan Motion Control IC User Guide*.

## 1.2 Prodigy/CME Stand-Alone Developer Kit

To facilitate initial system development and integration, PMD offers a developer kit for Prodigy/CME Stand-Alone Motion Boards. The Prodigy/CME Stand-Alone Developer Kit includes a complete Prodigy/CME Stand-Alone board, and the following software and accessory products are also included:

- Pro-Motion Windows-based exercisor software
- C-Motion SDK
- PDFs of all Prodigy documentation
- Various other cables and hardware accessories for connecting the board to external motion components

The table below provides additional information about the Prodigy/CME Stand-Alone Developer Kit that you may find useful:

DK Part Number	Prodigy/CME Stand-Alone Board Included	# Axes	Motors Supported
PRK13V58420	PR13V58420	4	Brushless DC DC Brush Step Motor (microstepping) Step Motor (pulse & direction)

## 1.3 Accessory Products

The Prodigy/CME Stand-Alone boards can be enhanced with the addition of any or all of the hardware accessory products described in the following table.

Component Part Number	Description
PW-5001-KIT-01.R	Power supply. This is a US Domestic & International wall adapter power supply and associated cable that plugs into the Prodigy/CME Stand-Alone board and provides power through its Power Connector.
Cable-5003-01.R	100-pin to dual 50 pin bifurcated ribbon cable. This 3 foot long cable connects to the primary motion connector (GP Connector) and provides two 50 pin header connectors suitable for connection to the IM-1000 breakout module (see below) or other compatible peripherals.

Component Part Number	Description
Cable-7003-01.R	60-pin to dual 30 pin bifurcated ribbon cable. This 3 foot long cable connects to the 'option' motion connector (Option Connector) and provides two 30 pin header connectors.
Cable-4355-01.R	Dual serial cable. This 5 foot long cable connects to the Prodigy/CME Stand-Alone'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 2 meter cable connects to the board's CANbus connector and has RJ45 connectors on both ends.
Cable-RJ45-02-R	Ethernet connector. This 2 meter cable connects to the board's Ethernet connector, and has RJ45 connectors on both ends.
DC-1000	Parallel encoder input adaptor. This daughter board module allows parallel-word and other encoders which use the SSI interface format to be directly connected.
IM-1000	Breakout interconnect module provides convenient jack-screw type terminators for the 100-pin cable. Used with Cable-Cable-5003-01.R.
IM-600	Breakout interconnect module provides convenient jack-screw type terminators for the 60-pin Option Connector. Used with Cable-7003-01.R.
Adapt-USB232-01.R	This adapter provides USB to serial conversion. It is useful for connecting to the Prodigy/CME's DB-9 serial port from a USB port.

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

## 1.4 Installation Overview

- 1 Before using the board, the software must be installed. See [Section 1.6, “Software Installation,”](#) for instructions on installing the software.
- 2 For a normal installation of a Prodigy/CME Stand-Alone board, you will need to configure the board for the specific motor hardware to which it will be connected. See [Section 1.7, “Preparing the Board for Installation,”](#) for a description of configuring the boards.
- 3 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 Stand-Alone board.
- 4 Connect the Prodigy/CME Stand-Alone 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 or standard operation of the board 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.
- 5 Once this hardware configuration is complete, the final step to finish the installation is to perform a functional test of the finished system. See [Section 1.10, “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.

## 1.5 Recommended Hardware

To install a Prodigy/CME Stand-Alone 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 pulse and direction, PWM, or analog-input amplifiers. The type of amplifier depends on the type of motor being used.
- 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, it's an option.
- Motion Connectors as required to connect the board to the amplifiers and the motors you have selected. The GP Connector cable (PMD p/n Cable-5003-01.R, 100 pin high density to dual 50-pin header-type connector) will be required, and for brushless DC motors as well as step motors driven in microstepping mode, the additional Option Connector cable (PMD p/n Cable-7003-01.R, 60 pin high density to dual 30-pin header-type connector) will be required. See [Section 4.3, "Connections Summary—Motor Amplifiers,"](#) for more information on setting up these connections.
- Power supply, power cable, and communication cables. From the PMD accessory cables list, this is p/n PW-5001-KIT-01.R power supply, Cable-4355-01.R Dual Serial Cable, and Cable-RJ45-02-R Ethernet Cable. See [Section 1.8.2, "Communication Connections,"](#) and [Section 1.8.3, "Power Connections,"](#) for a detailed description of how to connect these cables.

## 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

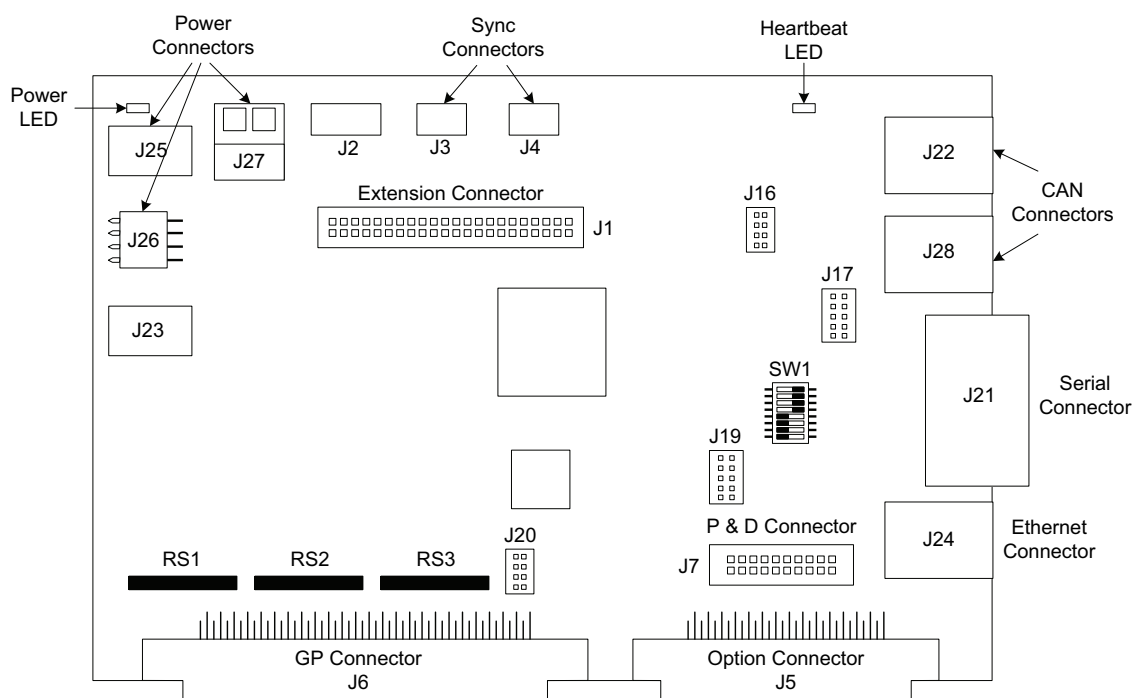
- Provided as a single DLL and Visual Basic .NET source code for easy porting onto various PC environments

VB Motion is documented in the *PMD Resource Access Protocol Programming Reference*.

## 1.7 Preparing the Board for Installation

Figure 1-1 shows the location of the resistor packs RS1, RS2, RS3, along with other components such as connectors. The front or top side of the board is shown, with the main- Prodigy/CME Stand-Alone connectors at the bottom. All connectors and user-adjustable components are located on the front side of the board. These items are listed in the table below.

**Figure 1-1:  
Prodigy/CME  
Stand-Alone  
Board  
Component  
Location**



### Board Components

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

Label	Description
RS1 - RS3	Resistor packs
SW1	Motor output configuration
J2	Reserved
J3, J4	Synch Connectors
J5	Option Connector
J6	GP Connector
J7	Pulse & Direction Connector
J16	Reserved
J17	Serial Alt. Connector
J19	Reserved
J20	Reserved
J21	Serial Connector
J22	CANbusI Connector
J23	Reserved

Label	Description
J24	Ethernet Connector
J27	Power Alt1 Connector
J26	Power Connector
J25	Power Alt2 Connector
J28	CANbus2 Connector

### 1.7.1 Resistor Pack Settings

The Prodigy/CME Stand-Alone 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 RS1, RS2, RS3	Installed; this is the default setting of resistor packs RS1 - RS3.	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 4.1.1.3, “Using Both Single-Ended and Differential Encoder Connections,”](#) for details.

### 1.7.2 Motor Output Configuration

There are two configurations for setting up the motor output pins on the Prodigy/CME Stand-Alone GP Connector: These motor output pins can be set up for use with DC Brush, Brushless DC, and microstepping motors, or the motor output pins can be set for pulse & direction motors. The default setting is for all motors to be set for DC Brush, Brushless DC, and microstepping.

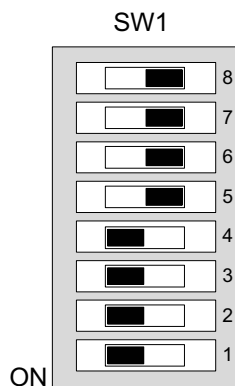
The switch bank SW1 controls these settings, which apply on a per-axis basis. The following table describes the correct switch setting for each output type. [Figure 1-2](#) provides a diagram of SW1, and [Figure 1-1](#) shows the location of SW-1 on the Prodigy/CME Stand-Alone board.

Axis	Set for DC Brush, Brushless DC, or Microstepping	Set for Pulse & Direction
1	1 on, 5 off	1 off, 5 on
2	2 on, 6 off	2 off, 6 on
3	3 on, 7 off	3 off, 7 on
4	4 on, 8 off	4 off, 8 on

Power to the board should be turned off when changing the state of the SW1 switches, and care should be taken not to set both switch pairs on (for example both 1 and 5 on, or both 2 and 6 on etc.) or damage may occur to the board.



**Figure 1-2:**  
**Switch settings**



## 1.8 Connection Summary

### 1.8.1 Motor Connections

The following sections summarize the recommended connections for various motor types. DC brush, brushless DC, microstepping, and step (pulse & direction) motors may be connected to the same board, so motor type is allowed to be different on a per-axis basis.

#### DC Brush Motors

The following table summarizes connections to the Prodigy/CME Stand-Alone boards when DC brush motors are used. Between one and four axes may be connected, depending on the specific Prodigy board and application requirements. All connections are made through GP Connector, J6 locatable on the board using [Figure 1-1](#). See [Chapter 4, Electrical Reference](#), for a detailed list of connections.

Signal Category	Signal Description
Encoder input signals: (per axis)	A quadrature Channel input B quadrature channel input Index pulse channel input
Amplifier output signals: (per axis, if PWM sign, magnitude used)	PWM direction PWM magnitude
Amplifier output signals: (per axis, if PWM 50/50 used)	PWM magnitude
Amplifier output signals: (per axis, if DAC output used)	DAC out
Other control signals: (optional per axis)	Home signal input Positive limit switch input Negative limit switch input AxisIn input AxisOut output
Miscellaneous signals:	Digital GND, AmpEnable, +5V (for encoder power)

#### Brushless DC Motors

The following table summarizes connections to the Prodigy/CME Stand-Alone boards when brushless DC motors are used. Between one and four axes may be connected, depending on the specific Prodigy board and application requirements. Connections are made through GP Connector, J6 and Option Connector, J5. Both of these connectors



are locatable on the board using [Figure 1-1](#). See [Chapter 4, \*Electrical Reference\*](#), for a detailed list of connections. See [Section 4.2.5, “Option Connector,”](#) for detailed information regarding the Option Connector.

Signal Category	Signal Description
Encoder input signals: (per axis)	A quadrature channel input B quadrature channel input Index pulse channel input
Amplifier output signals: (per axis, if PWM 50/50 used)	PWM magnitude (phase A) PWM magnitude (phase B) PWM magnitude (phase C)
Amplifier output signals: (per axis, if DAC output used)	DAC out (phase A) DAC out (phase B)
Hall inputs: (Option Con)	Hall (phase A) Hall (phase B) Hall (phase C)
Other control signals: (optional per axis)	Home signal channel input Positive limit switch input Negative limit switch input AxisIn input AxisOut output
Miscellaneous signals:	Digital GND, AmpEnable, +5V (for encoder power)

## Step Motors

The following table summarizes connections to the Prodigy/CME Stand-Alone boards when pulse & direction interface step motors are used. Between one and four axes may be connected, depending on the specific Prodigy board and application requirements. Unless differential pulse & direction output is desired, all connections are made through connector GP Connector, J6. This connector is locatable on the board using [Figure 1-1](#). See [Chapter 4, \*Electrical Reference\*](#), for a detailed list of connections.

Signal Category	Signal Description
Encoder input signals: (optional per axis)	A quadrature channel input B quadrature channel input Index pulse channel input
Amplifier output signals:	Pulse Direction
Other control signals: (optional per axis)	AtRest signal output Home signal channel input Positive limit switch input Negative limit switch input AxisIn input AxisOut output
Miscellaneous signals:	Digital GND, AmpEnable, +5V (for encoder power)

## Microstepping Motors

The following table summarizes connections to the Prodigy/CME Stand-Alone boards when microstepping-interface step motors are used. Between one and four axes may be connected, depending on the specific Prodigy board and application requirements. All connections are made through the GP Connector, J6, and Option Connector, J5. Both of these connectors are locatable on the board using [Figure 1-1](#). See [Chapter 4, \*Electrical Reference\*](#), for a detailed list of connections.

Signal Category	Signal Description
Encoder input signals: (optional per axis)	A quadrature channel input B quadrature channel input Index pulse channel input

Signal Category	Signal Description
Amplifier output signals: (per axis, if PWM sign, magnitude used)	PWM magnitude (phase A) PWM magnitude (phase B) PWM direction (phase A) PWM direction (phase B)
Amplifier output signals: (per axis, if PWM 50/50 used)	PWM magnitude (phase A) PWM magnitude (phase B)
Amplifier output signals: (per axis, if DAC output used)	DAC out (phase A) DAC out (phase B)
Other control signals: (optional per axis)	Home signal channel input Positive limit switch input Negative limit switch input AxisIn input AxisOut output
Miscellaneous signals:	Digital GND, AmpEnable, +5V (for encoder power)

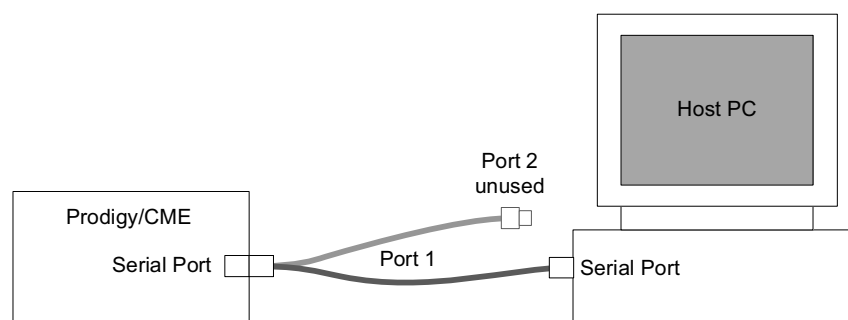
## 1.8.2 Communication Connections

While the Prodigy/CME Stand-Alone 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](#), and the *Pro-Motion User Guide*.

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 Stand-Alone board’s J21 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 1-3](#) shows a typical serial port connection.

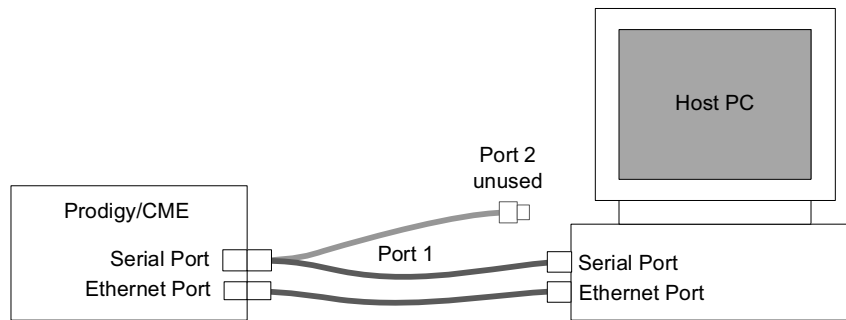
**Figure 1-3:**  
Serial Port  
Connection



The Ethernet connection will not be made until serial communications are established. See [Section 1.10.2, “Changing the Ethernet Parameters of the Prodigy/CME Stand-Alone Board,”](#) 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 1-4](#) shows the complete serial and Ethernet connection setup.

The communication connections are locatable on the board using [Figure 1-1](#).

**Figure 1-4:  
Serial and  
Ethernet  
Connection**



### 1.8.3 Power Connections

There are a number of options for providing power to the Prodigy/CME Stand-Alone board. For first time system verification, however, the convenient AC wall adapter power supply and cable set contained in PMD accessory p/n PW-5001-KIT-01.R will be used. Plug the AC wall adapter into any AC outlet, and connect the power plug to J25 on the board. This connector is locatable on the board using [Figure 1-1](#). Outside of the US & Canada, one of the AC adapter plugs included with the power supply will be needed depending on the plug type used with your AC service.

In addition to the power supply and connections for the Prodigy/CME Stand-Alone board, the motor amplifiers also need to have power provided to them. Consult the technical manual for the amplifiers you are using for more information. While it is possible for the motor power to be the same as the board power (which is 5V), it is much more likely that the power provided to the amplifiers will be at a higher voltage, and therefore be provided by a different power supply.

## 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 Stand-Alone's green power LED should light. This LED is locatable using [Figure 1-1](#). If the LED does not light, recheck the connections.

After power up, the board will be in a reset condition. In this condition, 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 amplifier and encoder connections. If anomalous behavior is still observed, call PMD for assistance. Complete PMD contact information is listed on the last page of this manual.

When using a PR13x58x20 board there are some combinations of motor output signal and amplifier type where undesired motion may occur when the board is powered up. In particular, if the connected amplifier is a PWM 50/50 amplifier, the motor will receive 100% power because the PR13x58x20 Prodigy/CME Stand-Alone board defaults to a different PWM signal encoding, PWM sign/magnitude, on reset. To avoid this situation, use AmpEnable as an enable/disable signal for the amplifier, and set the motor type for each axis before enabling the amplifier. See [Chapter 2, Operation](#), for more information on this function.





To minimize the risk of unintended machine motion during powerup, care should be taken to control the overall powerup sequence of the Prodigy/CME board and connected system motion amplifiers. The recommended sequence is to first powerup the Prodigy/CME board, followed by powerup of the motion amplifiers. The time delay to ensure stable operation of the control electronics before powering up the amplifiers varies depending on the system's power supplies and configuration, but a typical safe figure is 500 mSec - 1,000 mSec.

## 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 Stand-Alone 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.

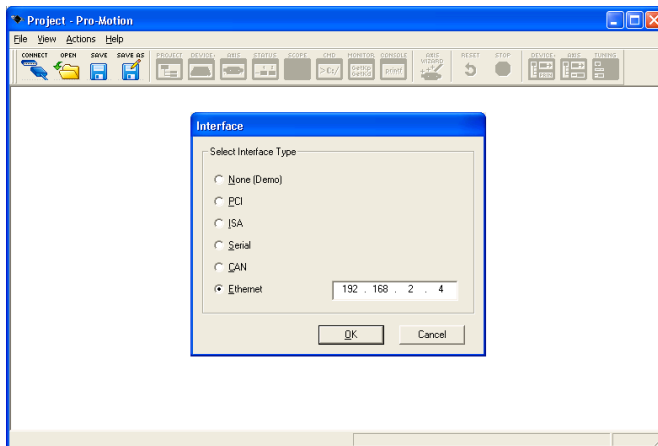
During this first time system setup you may find it useful to refer to other PMD manuals such as the *Magellan Motion Control IC User Guide* to familiarize yourself with operation of the Magellan Motion Control IC, which lies at the heart of all Prodigy Motion boards.

### 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.

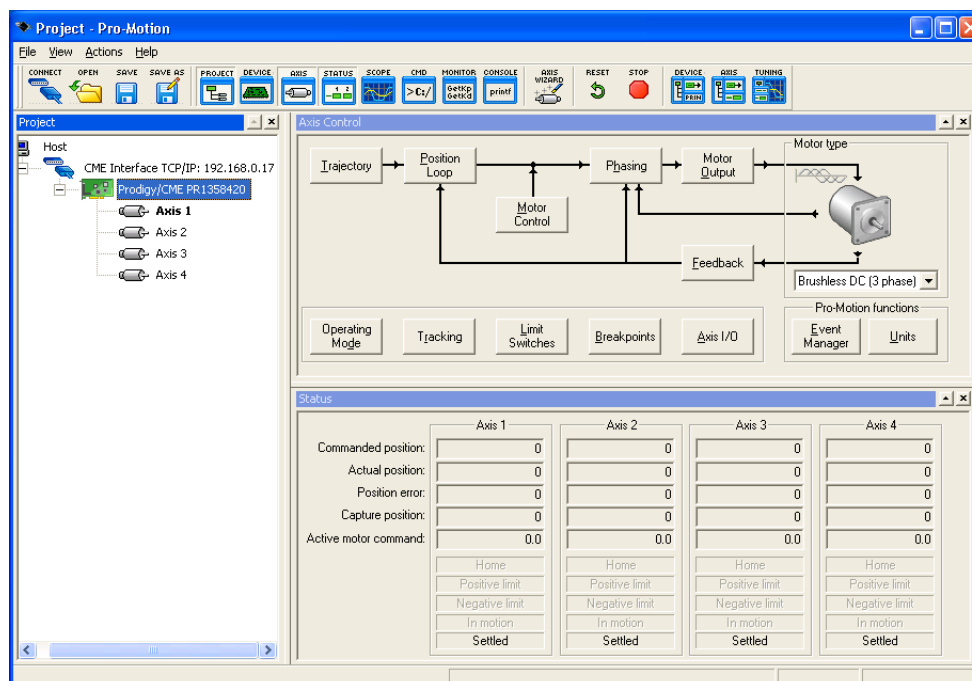
The purpose of the Interface dialog box is to indicate to Pro-Motion how your Prodigy/CME Stand-Alone board is connected to the PC. It provides various selectable communication options such as PCI, serial, CANbus, Ethernet.

- 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 10 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 Changing the Ethernet Parameters of the Prodigy/CME Stand-Alone Board

To change the Ethernet parameters:

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

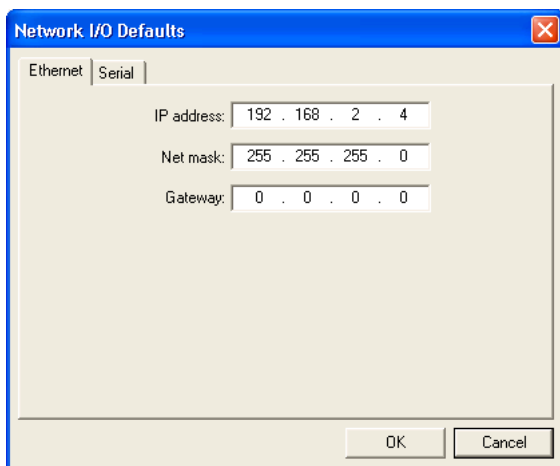
The Device window appears.

- 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 1.8.2, "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 Stand-Alone 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, "Changing the Ethernet Parameters of the Prodigy/CME Stand-Alone Board,"](#) 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 Stand-Alone 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 Stand-Alone 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 Stand-Alone board. 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, “Changing the Ethernet Parameters of the Prodigy/CME Stand-Alone Board,”](#) for a description of changing the IP address.

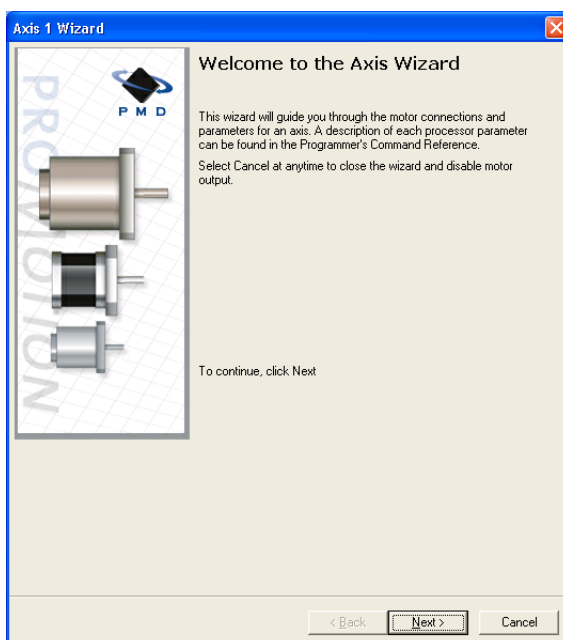
## 1.10.5 Initializing Motion Axes

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. For step motors this means the pulse & direction connections are working properly, and for servo motors this means the encoder and amplifiers connections have been validated, and stable (but not necessarily optimal, see caution below for more information) servo tuning parameters have been loaded into the Prodigy/CME board's Magellan Motion Control IC. Depending on the signals connected, this may also mean that limit switches, and other hardware connections are functioning properly.

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.

- 3 Click the Trajectory button in the Axis Control window.

The Trajectory dialog box appears.

- 4 In the Profile mode list, select Trapezoidal.



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

Pro-Motion provides various selectable units for distance and time, but defaults to units of encoder counts (or pulses for step motors) for distance and seconds for time. This means the default units for velocity are counts/sec, and the default units for acceleration and deceleration are counts/sec<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

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

Congratulations! First time system verification for this axis is now complete. You should now initialize all of the axes in your system. Go to [Section 1.10.5, “Initializing Motion Axes,”](#) and repeat the steps.

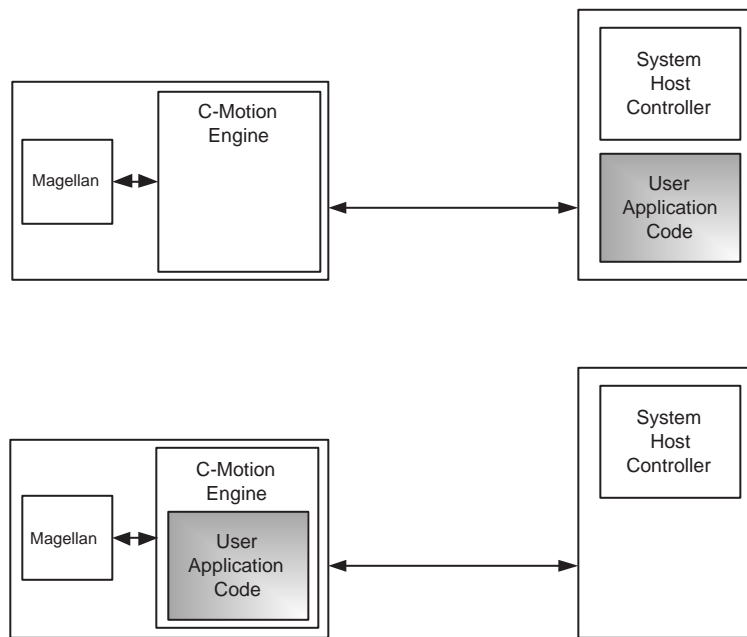
## 1.11 Developing User Application Code

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

For more information on VB-Motion and C-Motion, please consult the *Magellan Motion Control IC Programming Reference*.

### 1.11.1 Architecture

[Figure 1-5](#) shows two ways to locate your application code with the Prodigy/CME Stand-Alone boards.



**Figure 1-5:**  
**Two Ways to**  
**Locate the**  
**Code on the**  
**Prodigy/CME**  
**Stand-Alone**  
**Board**

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

Either VB-Motion 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 Stand-Alone 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 machine 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 stand-alone 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 his machine, and locating his software on the hardware platform that will best match his 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 Stand-Alone'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 Stand-Alone board.

## 2. Operation

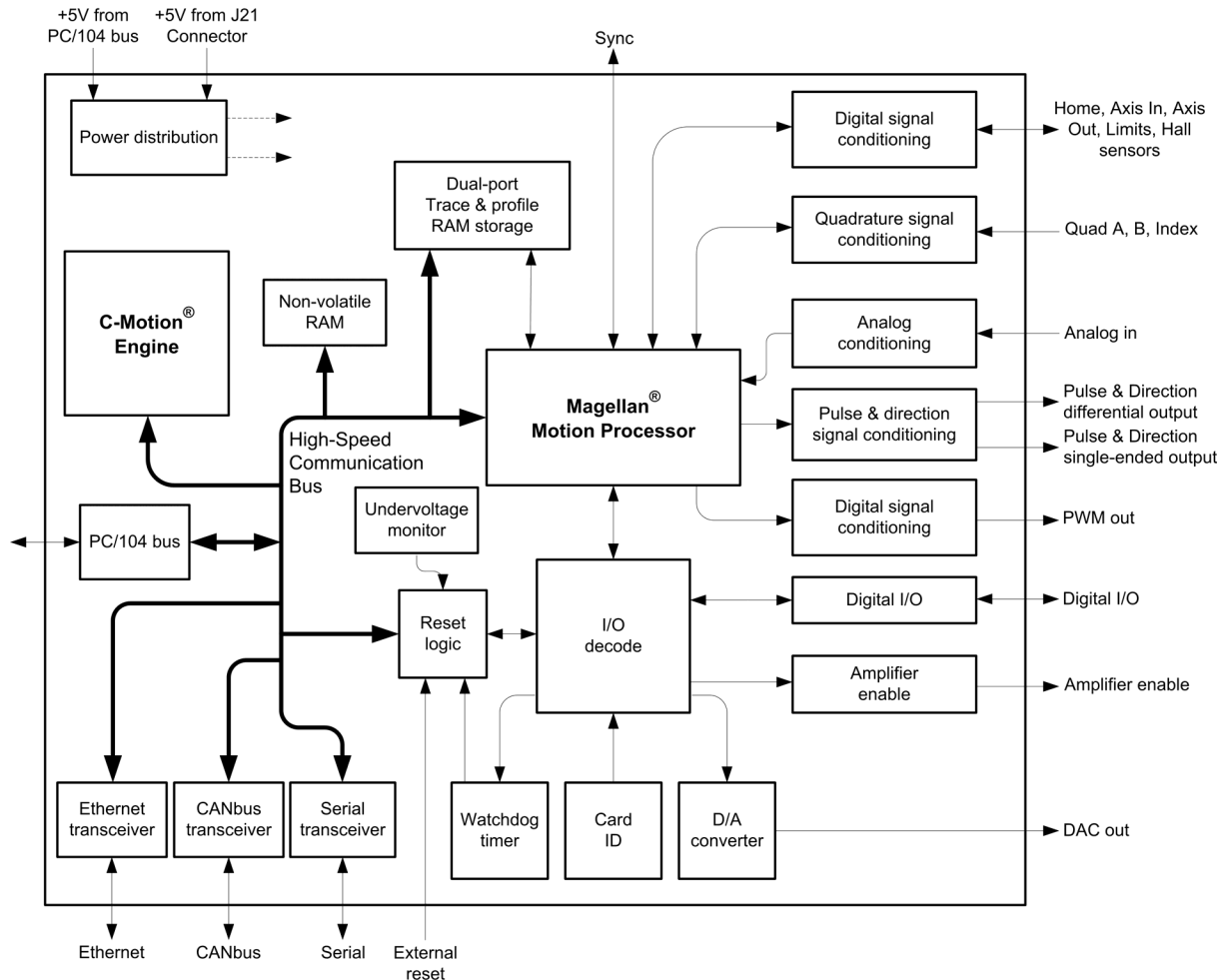
### ***In This Chapter***

- ▶ Board Function Summary
- ▶ Magellan Motion Control IC Functions
- ▶ Magellan-Controlled Functions
- ▶ C-Motion Engine Functions
- ▶ Communications Functions
- ▶ General Board Functions
- ▶ Signal Processing and Hardware Functions
- ▶ Software Libraries

The Prodigy/CME Stand-Alone products are high-performance board-based motion controllers for DC brush, brushless DC, pulse & direction, and microstepping motors. These boards are based on Magellan Motion Control ICs, which perform motion command interpretation and numerous other real-time functions. The ‘/CME’ series Prodigy boards include a powerful C-Motion Engine module which allows C-Motion application code to be downloaded and executed directly on the motion board, improving run time performance, enabling distributed control architectures, and allowing a complete machine controller to be hosted on a single board.

The following diagram provides a functional block diagram of the Prodigy/CME Stand-Alone board:

**Figure 2-1:  
Prodigy/CME  
Stand-Alone  
Board Internal  
Block Diagram**



## 2.1 Board Function Summary

The Prodigy/CME Stand-Alone board functions can be broken down into six overall categories:

**Magellan Motion Control IC functions** - These are user-accessible functions which reside in the Magellan Motion Control IC. Included are profile generation, DC brush and brushless DC loop closure, microstep generation 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 related hardware.

**Magellan-controlled functions** - These are user-accessible functions which are controlled by the Magellan Motion Control IC, but which reside outside the Magellan chip on various portions of the board circuitry. These functions include general purpose digital I/O, a watch-dog timer, and many other capabilities.

**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 Stand-Alone 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 peripherals such as the dual-ported RAM.

**General board functions** - These are user-accessible general purpose board resources not specifically controlled by the Magellan Motion Control IC. Examples include board reset, board default parameter restore, and dual-ported and non-volatile RAM access.

**Signal processing & hardware functions** - A substantial portion of the board performs signal conditioning and other functions associated with safety-related signal processing. These functions are a fixed feature of the board and are not user-accessible.

**Communications functions** - The Prodigy/CME Stand-Alone board provides sophisticated resource addressing capabilities which allow an external host controller, or the onboard C-Motion Engine, to address various on-board and board-connected resources. In addition, various direct management features are provided for the Prodigy/CME Stand-Alone board's communication ports (two serial ports, a CANbus port, and an Ethernet port).

Section 2.1.1, “[Board Access Basics](#),” provides an introduction to general board access issues that will be useful while reading the remainder of this chapter.

Section 2.2, “[Magellan Motion Control IC Functions](#),” through Section 2.7, “[Signal Processing and Hardware Functions](#),” describe each of these specific functional areas.

Section 2.8, “[Software Libraries](#),” provides an overview of accessing Prodigy/CME Stand-Alone board functions via software libraries.

## 2.1.1 Board Access Basics

Access to the Prodigy/CME Stand-Alone 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 Stand-Alone 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 Stand-Alone 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 Stand-Alone 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 Stand-Alone 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.

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.

[Chapter 3, Accessing Board Resources](#), describes all of these constructs in more detail. In the subsequent sections of this chapter a summary of the PRP actions that are required to access each board function are included with the descriptions of the board functions themselves.

## 2.2 Magellan Motion Control IC Functions

The Magellan Motion Control IC block pictured in [Figure 2-1](#) forms the core of Prodigy Stand-Alone boards. Here is an overview of the functions provided, or managed, by the Magellan Motion Control IC on the Prodigy/CME Stand-Alone boards:

- Profile generation
- Motor output signal generation (PWM, analog, and pulse & direction)
- 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 at-settled indicator
- Limit switches
- Access to various on-board resources such as parallel I/O and dual-port trace & profile RAM

The Magellan Motion Control IC interfaces with motion hardware components such as feedback encoders, motor out signal generation hardware, and others, directly through its own pin connections, and through various signal conditioning circuitry. See [Section 2.7, “Signal Processing and Hardware Functions,”](#) for a complete description of on-board hardware interconnections and signal management.

The Magellan instruction set is very flexible and powerful. The following example, which would be used to set up and execute a simple trapezoidal profile, illustrates just a small part of the overall command set:

```

SetProfileMode AxisI, trapezoidal    // set profile mode to trapezoidal for axis I
SetPosition AxisI, 12345             // load a destination position for axis I
SetVelocity AxisI, 223344            // load a velocity for axis I
SetAcceleration AxisI, 1000          // load an acceleration for axis I
SetDeceleration AxisI, 2000          // load a deceleration for axis I
SetUpdateMask AxisI, Profile         // specify that an update of profile parameters only
                                     // is to occur
Update AxisI                         // Double buffered registers are copied into
                                     // the active registers, thereby initiating the move

```

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. Within the Prodigy/CME Stand-Alone board, the Magellan uses its high-speed parallel-word communications mode to connect to the board's 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 Stand-Alone board by serial, CANbus, or Ethernet port.

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 Reference*. These documents also describe VB-Motion, and C-Motion, which are the software libraries that are used to send commands to the Magellan chip and exercise its many functions.



## 2.2.1 Accessing the Magellan Motion Control IC

To send and receive command packets to the Magellan Motion Control IC the PRP action **Command** is used. The Magellan command packet is loaded into the PRP message body, and the return PRP message body contains the return packet provided by the Magellan. A return without error indicates that the command was processed successfully. If an error occurred while the Magellan was processing the command, the message body is loaded with the specific error that occurred. For more information on Magellan command packet formats and return packet formats see the *Magellan Motion Control IC Programming Reference*.

In addition to accessing the on-board Magellan, it is also possible to access Magellan Motion Control ICs that are connected via the Prodigy/CME Stand-Alone board's attached networks such as the CANbus, Serial, or Ethernet networks. An additional PRP action, **Open**, sent to either the **Device** or the **Peripheral** resource, is used to establish a connection to such a motion control IC. See [Section 3.4, "Accessing Magellan-Attached Devices,"](#) for more information on connecting to motion control ICs on attached networks.

For complete information on the format and function of these, and other PRP actions, refer to the *Prodigy/CME Programmer's Reference*.

### 2.2.1.1 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. The PRP action **Reset**, when sent to the MotionProcessor resource, causes a reset of the Magellan Motion Control IC. This reset affects the Magellan Motion Control IC, and a number of Magellan-controlled signals. Note however that this type of reset is different than a full board reset initiated via the PRP action **Reset**. See [Section 2.6.2, "Reset,"](#) for a description.

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

Signal Name	State
AxisOutI-4	High
PWMMagIA-4C	Low
PWMSignIA-4B	Low
DACIA-DAC4B	No change
DigitalOut0-7	No change
AmpEnableI-4	No change
DAC On/Off	No change
Watchdog Timer	No change

## 2.2.2 Connections & Associated Signals

The Prodigy/CME Stand-Alone board's Magellan Motion Control IC has extensive signal connections associated with it to allow interfacing to various motion peripherals. See the *Magellan Motion Control IC User Guide* as well as [Section 2.7, "Signal Processing and Hardware Functions,"](#) for a complete description of the signals managed by the Magellan Motion Control IC.

## 2.3 Magellan-Controlled Functions

There are a number of motion control hardware features on the Prodigy/CME Stand-Alone board that are controlled through the Magellan Motion Control IC, but are not located within the Magellan Motion Control IC itself. Because of this they are not specifically described in the *Magellan Motion Control IC User Guide*, or the *Magellan Programming Reference*, instead being described here, in subsequent sections.

These Magellan-controlled board functions are:

- General Purpose Digital I/O
- Amplifier Enable
- DAC Output Enable
- Watchdog Timer
- Undervoltage Monitor
- Reset Monitor
- Board ID
- Dual-Port Trace & Profile RAM

These motion hardware control functions are accessed via the Magellan's **ReadIO** and **WriteIO** commands, which provide control of hardware peripherals external to the Magellan Motion Control IC. In addition, many of the Prodigy Stand-Alone board motion hardware control functions can be accessed using VB-Motion and C-Motion 'named' commands specific to that function. See the subsequent feature-specific sections for a description of both access methods.

The **ReadIO** and **WriteIO** commands are similar to any other Magellan packet commands in that they are accessed via the PRP action command. See [Section 2.2.1, "Accessing the Magellan Motion Control IC,"](#) for detailed information.

### 2.3.1 General-Purpose Digital I/O

In addition to numerous special-purpose digital signals which are input or output to the board, and which are directly processed by the Magellan Motion Control IC such as *AxisIn*, *AxisOut*, *Home*, and *QuadA*, the Prodigy/CME Stand-Alone boards also support eight general-purpose inputs, and eight general-purpose outputs. These signals provide a convenient way of accessing additional application-specific general-purpose digital I/O.



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

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

The eight inputs and eight outputs are read using the Magellan Motion Control IC's **ReadIO** command and written using the **WriteIO** command, with an I/O address of 0. This is illustrated in the following table, along with the bit locations of the input and output signals.

I/O Address	Bit Location	Signals
0	0 - 7	DigitalOut0-7
	8 - 15	DigitalIn0-7

To read all eight general-purpose digital I/O signals, a **ReadIO** command is performed at address offset 0. The 16-bit read word returns the current output values (set using the **WriteIO** command) in bits 0 - 7, while bits 8 - 15 hold the digital values corresponding to the signal levels at the connector for those inputs. To write new signal values to the eight digital outputs, a **WriteIO** command to address offset 0 is sent, and the values on bits 0-7 will be output to the signal connections. The values of bits 8 - 15 are ignored.

### Example

To write the value 0xAA to bits 0 - 7, the command **WriteIO** is used. If the signal pattern 0x55 is present on the eight input connections, then the command **ReadIO 0** will return the value 0x55AA. The upper eight bits reflect the present value of the input signals, while the lower eight bits reflect the 8-bit value being output.

In addition to the low-level **ReadIO** and **WriteIO** commands, the following commands are also supported in C-Motion and VB-Motion: **WriteDigitalOutput**, **ReadDigitalInput**, and **ReadDigitalOutput**. These commands provide a clearer and simpler interface to the Prodigy/CME Stand-Alone board's general purpose I/O signals by handling the byte shifting.

#### 2.3.1.1 Connections & associated signals

The general-purpose I/O are direct digital inputs and outputs. There are no associated connections required for these signals to function properly, however, one or more of the digital grounds must be connected. Digital inputs are pulled up through 4.7 kOhm resistors to 5V. The power-up default value for all general-purpose digital outputs is low.

See [Chapter 4, Electrical Reference](#), for a complete description of the pinout connections to and from the board.

### 2.3.2 Amplifier Enable

The signals **AmpEnable 1-4** provide four digital outputs which may be used as amplifier enable signals. They can also be used as general-purpose digital outputs. They can be read or written using the Prodigy/CME Stand-Alone board Magellan's **ReadIO** and **WriteIO** commands.

These outputs are read using the **ReadIO** command, and written to using the **WriteIO** command, using an address of 1, as shown in the following table:

I/O Address	Bit Location	Signals
1	0-3	Amplifier enable outputs (0-3)
	4-6	Unused
	7	DAC enable status (1 = enabled; 0 = disabled)
	8-11	Change mask for bits 0-3; amplifier enable outputs (1 = change; 0 = don't change)
	12-14	Unused
	15	Change mask for DAC enable (1 = change; 0 = don't change)

To read the status of the amplifier enable outputs, the command **ReadIO** is used at address 1. The values currently being output will appear in bits 0 - 3. To write values to the amplifier enable output signals, the **WriteIO** command is used with an address of 1. An amplifier enable signal will be changed only if the corresponding change mask bit is set. The change mask bits are bits 8 - 11, the values to be loaded are bits 0 - 3.

#### Examples

```
WriteIO Address1, 0x0505      // write 0x0505 to I/O Address 1 to enable Amplifiers 1 & 3, don't change 2 & 4
WriteIO Address1, 0x0400      // write 0x0400 to I/O Address 1 to disable Amplifier 3, don't change 1, 2, & 4
WriteIO Address1, 0x0100      // write 0x0100 to I/O Address 1 to disable Amplifier 1, don't change 2, 3, & 4
```

In addition to the low-level **ReadIO** and **WriteIO** commands, the following commands are also supported in C-Motion and VB-Motion: **SetAmplifierEnable** and **GetAmplifierEnable**. These commands provide a clearer and simpler interface by handling the bit shifting.

### 2.3.2.1 Connections & associated signals

*AmpEnable1-4* are direct digital outputs. There are no associated connections required for these signals to function properly, however, one or more of the digital grounds must be connected. The power-up default value for all amplifier enable signals is low (disabled). See [Chapter 4, Electrical Reference](#), for a complete description of the pinout connections to and from the board.

### 2.3.3 DAC Output Enable

In addition to the amplifier enable outputs, there is a dedicated board function which allows the DAC output signals to be shunted to 0 volts for safety purposes (DAC disabled), or to be controlled by the Magellan Motion Control IC (DAC enabled). This shunting occurs at a hardware level outside the motion control IC itself, and provides an additional safety layer to control the motor command.

The status of the DAC output enable function can be read using the **ReadIO** command, and the DAC output enable status can be set using the **WriteIO** command, with an address of 1. The table in [Section 2.3.2, “Amplifier Enable,”](#) shows this. To read the status of the DAC output enable function, **ReadIO** is used. The value currently in use will appear in bit 7. A value of 1 indicates DAC output is enabled, meaning that the voltage being output by the DACs is controlled by the motion control IC. A value of 0 indicates that it is disabled, meaning that the voltage being output by the DAC is forced to 0.0 volts.

To enable or disable the DAC enable function, the **WriteIO** command is used. The change mask bit located at bit 15 must be loaded with a 1. Bit 8 must be loaded with a value of 0 to disable, or a value of 1 to enable output.

The powerup default value for DAC Output Enable is disabled. In addition, the DAC Output Enable is disabled upon a board reset, or via the external **Reset** signal. See [Section 2.6.2, “Reset,”](#) for more information.

#### Examples

```
WriteIO Address1, 0x8080           // write 0x8080 to I/O Address 1 to enable all DACs,
                                   // don't change amplifier enables
WriteIO Address1, 0x8000           // write 0x8000 to I/O Address 1 to disable all DACs,
                                   // don't change amplifier enables
```

In addition to the low-level **ReadIO** and **WriteIO** commands, the following commands are also supported in C-Motion and VB-Motion: **SetDACOutputEnable** and **GetDACOutputEnable**. These commands provide a clearer and simpler interface by handling the bit shifting.

### 2.3.4 Board Watchdog Timer

To enhance the overall safety of the board, a watchdog function has been included. When enabled, the watchdog will automatically trigger a board reset if communication to it should be lost. See [Section 2.6.2, “Reset,”](#) for a description of the actions and signal changes that occur after a board reset.

To enable the board watchdog timer, the **WriteIO** command is used to send a value of 0x5562 to address 4. Once enabled, the watchdog timer will time out, causing a hard reset, if another write to address 4 with a value of 0x5562 is not received within 104 mSec. As long as a watchdog value is written to address 4 within the 104-mSec interval, no reset will occur and motion operations will proceed normally. Once enabled, the watchdog mechanism cannot be stopped until a reset or power cycle occurs.

After powerup or board 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 and triggers a reset, it also disables itself.

In addition to the low-level **ReadIO** and **WriteIO** commands, the following command is also supported by C-Motion and VB-Motion: **SetWatchDog**. This command provides a clearer and simpler interface by automatically sending the value 0x5562.

## 2.3.5 Undervoltage Monitor

To enhance reliability under a variety of electrical conditions, an undervoltage detection circuit has been included. This circuit triggers a hard reset when the voltage has dropped to an unsafe level. Resetting the Prodigy/CME Stand-Alone board will have the result of setting all motor command outputs to zero, thus allowing the motors to come to a safe stop. An undervoltage condition is detected when the 3.3V internal supply on the board drops below 2.7V. See [Section 2.3.6, “Reset Monitor,”](#) to determine if a reset was caused by an undervoltage condition.

## 2.3.6 Reset Monitor

During normal operations, the Prodigy/CME Stand-Alone board is only reset during power-up. There are, however, several other ways that the Prodigy/CME Stand-Alone board can be reset. See [Section 2.6.2, “Reset,”](#) for a complete description of how the board can be reset. A reset serves the purpose of initializing values and bringing the Prodigy/CME Stand-Alone board to a known and consistent state.

To determine the cause of a board reset, special instructions to read the reset source have been provided. The command **ReadIO** with an address of 2 should be used. The following table details the encoding of this I/O address word.

I/O Address	Bit Location	Signals
2	0-10	<i>Reserved</i>
	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 the PRP action Reset.
	13	Undervoltage detection: a 1 value in this bit indicates a reset caused by undervoltage detection.
	14	External signal: a 1 value in this bit indicates a reset caused by the external <b>Reset</b> signal, pin 91 of the GP Connector (J6)
	15	Board Watchdog timeout: a 1 value in this bit indicates a reset caused by the board watchdog timeout. See <a href="#">Section 2.3.4, “Board Watchdog Timer,”</a> for a description.

Once a reset condition has occurred, the reset status stored at address 2 (described in the preceding table) can be cleared by a **WriteIO** command to address 2 with a value of zero (0).

### Example

To determine that a reset has occurred, and to determine the cause of the reset, the command **ReadIO** is used. Assuming that a watchdog timer event has occurred, the value returned would be 0x8000. To clear the reset monitor word, the command **WriteIO** is sent to address 2 with a value of zero (0).

In addition to the low-level **ReadIO** and **WriteIO** commands, a **GetResetCause** command is also supported by C-Motion and VB-Motion. This command returns the cause and also clears the reset condition.

## 2.3.7 Board ID

This feature allows the user to query the board for a Board ID. This may be helpful for verifying the type of Prodigy Motion Board in situations where multiple boards of varying types are installed.

### 2.3.7.1 ReadIO and WriteIO commands

To read the Board ID, the **ReadIO** command is used with an address of 0xFF. The encoding of the bits returned is detailed in the following table:

Address	Bit Location	Signals
0xFF	0-3	Major board revision: This nibble encodes the major board revision. This value can range from 0 to 15.
	4-7	Minor board revision: This nibble encodes the minor board revision. This value can range from 0 to 15.
	8-11	Board generation: This nibble encodes the board generation. This value can range from 8 to 15. (0 to 7 are reserved for older motion board families.)
	12-14	Board type: This nibble encodes the board type and has one of the following values: 0 = ISA Bus 1 = PCI Bus 2 = CompactPCI 3 = PC/104 4 = MIPS 5 = RS232 6 = CAN 7 = Standalone
	15	0 = Standard Prodigy board 1 = Prodigy/CME Stand-Alone board

#### Example

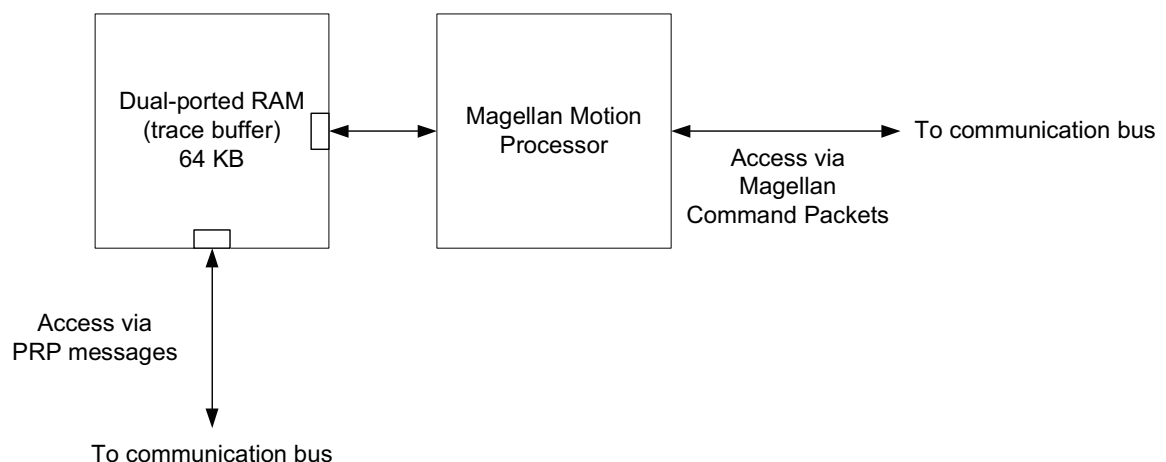
To read the Board ID, the command **ReadIO** is used. For example, the value 0xf805 would be interpreted as: Prodigy/CME Stand-Alone board, board generation 8, board revision 5.0.

In addition to the low-level **ReadIO** command, a **ReadboardID** command is also supported by C-Motion and VB-Motion. This command returns the Board ID in the format described above.

### 2.3.8 Dual-Ported RAM (Trace Buffer)

The Prodigy/CME Stand-Alone board has 64 KBytes of on-board dual-ported memory (DPRAM) which has one 'port' interfaced to the Motion Control IC, and the other 'port' interfaced to the board's high speed internal communications bus, allowing two paths of communication. [Figure 2-2](#) shows this configuration.

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



The dual-ported RAM trace buffer is a powerful feature that allows 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 Stand-Alone 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*.

### 2.3.8.1 Accessing the dual-ported RAM

To access the contents of the trace buffer 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 the trace buffer after it has been written to by the Magellan Motion Control IC is via the Prodigy/CME Stand-Alone board's high speed communication bus. In this mode the dual-ported RAM is referred to as a resource, and is accessed via the PMD Resource Access Protocol. See [Section 2.6.4, "Dual-ported RAM \(Trace Buffer\),"](#) for more information on this access method.

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



## 2.4 C-Motion Engine Functions

The C-Motion Engine on the Prodigy/CME Stand-Alone 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.

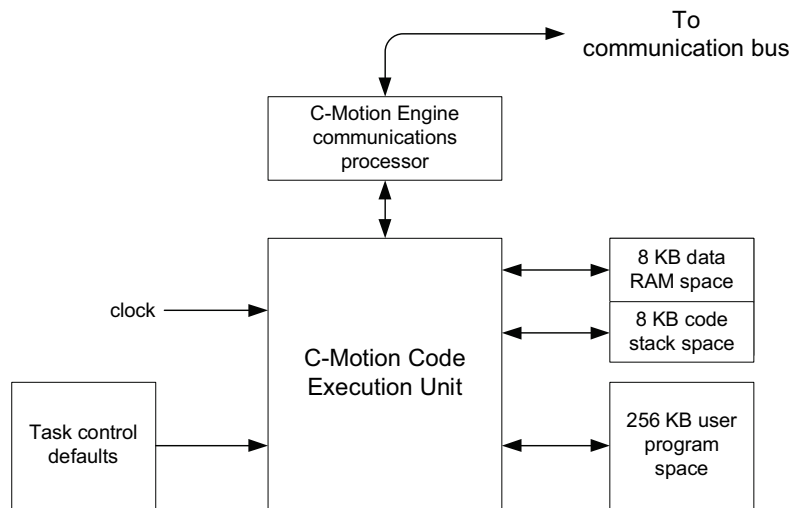
### 2.4.1 C-Motion Engine Hardware Configuration

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 Stand-Alone 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



**Figure 2-3:  
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

## 2.4.2 Powerup & Operation

Upon reset or power up 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 run time 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 Stand-Alone board. This C-Motion Engine-initiated reset is identical to the reset that occurs after sending a PRP **Reset** action, except that the cause of the reset is recorded as ‘C-Motion Engine user code Fault’ rather than ‘commanded’ reset. See [Section 2.6.2, “Reset,”](#) for more information on the Reset command. See [Section 2.3.6, “Reset Monitor,”](#) for information on retrieving the reset cause.

Whether or not user application code is running, after reset or power up, 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 onto 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.

### 2.4.3 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 power up, 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 PRP action **Command** is sent to the **CMotionEngine** resource.

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 power up or reset. The PRP action **Set** sent to the **CMotionEngine** resource allows setting of the user code start mode.

In addition to these functions, it is also possible to determine whether the user application code is presently running or not. This status information may be useful during code debugging, or to help diagnose problems. This capability is accessed via a **Get** action sent to the **CMotionEngine** resource.

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

### 2.4.4 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 Robot Arm” 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 Stand-Alone board's peripheral mechanism described in [Section 3.2.1, “Peripheral Connections,”](#) 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 PRP **Send** or **Receive** action is sent to the **CMotionEngine** resource. To receive these messages within the C-Motion Engine a special ‘user packet’ peripheral is opened. For more information see the *Prodigy/CME Programmer’s Reference*.

In addition to these communications commands, when sent to the **CMotionEngine** resource, the PRP action **NOP** performs a basic connection check. The message body is empty. A return without a PRP error code indicates that the C-Motion Engine is accessible and processing commands.

### 2.4.5 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 Stand-Alone 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 Stand-Alone 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.

### 2.4.6 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 Serial 2, however this can be changed using the PRP action **Set** with the **Device** resource specified as CANbus or Ethernet/UDP.

### 2.4.7 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 PRP action **Get** is used, specifying a resource ID of **CMotionEngine**. 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 PRP actions, refer to the *Prodigy/CME Programmer's Reference*.

## 2.4.8 C-Motion Engine Heartbeat LED

The Prodigy/CME Stand-Alone board utilizes an LED, locatable using [Figure 1-1](#), 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.

## 2.5 Communications Functions

The Prodigy/CME Stand-Alone board provides three different network connection types, Serial, CANbus, and Ethernet. Access to these communication resources is provided via a peripheral connection. A peripheral is a resource, and is utilized by various PRP actions to send and receive messages to network connections.

Basic access to either the Serial, CANbus, or Ethernet port is accomplished by opening a peripheral with the detailed connection parameters that will be used during communications associated with that peripheral connection, and then sending and receiving messages via that peripheral resource address.

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

In the subsequent sections the operational characteristics of the Serial (Serial 1 & Serial 2), CANbus, and Ethernet network ports will be detailed. See [Section 3.2.1, "Peripheral Connections,"](#) for more information on the **Peripheral** resource and how it is used.

### 2.5.1 Serial 1 & Serial 2

CME-Standalone Motion Boards provide asynchronous serial communications in either RS232 or RS485 mode. Access to the serial port controller is managed using peripheral connections. See [Section 3.2.1, "Peripheral Connections,"](#) for more information on creating and using peripheral connections.

In RS232 mode two serial ports are supported, referred to as Serial 1, and Serial 2. 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 Serial 1. Also in RS485 mode, the serial port may be operated in either half duplex or full duplex mode.

Pin #1 of the J21 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 takes effect only 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 Serial 1 and Serial 2:

Parameter	Range	Serial 1 Default	Serial 2 Default
Baud rate	1200 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

To create a serial port peripheral connection, the above parameters are specified in the PRP action **Open** with the resource set to **Device**. Messages to and from the Serial port are transmitted via the **Send** and **Receive** actions sent to the **Peripheral** resource. Whenever a new serial port peripheral is opened its previous function is canceled. By default Serial 1 listens for PRP communications and Serial 2 is the console port.

After a reset or at power-up, the board retrieves default information for the serial ports. To change these default values, the PRP action **Set** is send with a resource of **Device**. [Section 2.6.3, "Setting Board Defaults,"](#) describes this further.

For detailed information on PRP action formats and function, refer to the *Prodigy/CME Programmer's Reference*.

### 2.5.1.1 Connections & associated signals

J21 comprises a special 9-pin connector used to connect one or both serial ports. See [Section 1.3, "Accessory Products,"](#) and [Section 4.2.6, "Serial Connectors,"](#) for a detailed signal description of the Serial connector.

## 2.5.2 CANbus Communications

The Prodigy/CME Stand-Alone board provides a general purpose CANbus port compatible with the CAN 2.0B standard which may be operated at various communication rates from 10,000 to 1,000,000 bps (bits per second). In addition, each CANbus devices is assigned two CAN identifiers (also called a 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

To create a CANbus peripheral conversation, the baud rate, send address and receive address are specified in the PRP message body of an **Open** action sent to the **Device** resource. Messages to and from the CANbus port are transmitted via the **Send** and **Receive** actions sent to the **Peripheral** resource.

After a reset or at power-up, the board retrieves default information for the CANbus port. To change these default values, the PRP **Set** action is used with a resource of **Device**. See [Section 2.6.3, "Setting Board Defaults,"](#) for more information.

For detailed information on PRP action formats and function, refer to the *Prodigy/CME Programmer's Reference*.

### 2.5.2.1 Connections & associated signals

J22 and J28 provide standard RJ-45 connectors to connect to a CANbus network. See [Section 4.2.8, “CAN Connectors,”](#) for a detailed signal description of the CAN connectors.

## 2.5.3 Ethernet Communications

The Prodigy/CME Stand-Alone 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 2.4.6, “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 net mask 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 Stand-Alone board, TCP rather than UDP communications are used. To determine what the unused IP addresses are for your Ethernet network, and what values for net mask and gateway to use, you should contact your network administrator.

By convention, the 32 bit values for IP Address, Net mask, 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 Stand-Alone board:

Parameter	Range	Default
IP address	0.0.0.0 – 255.255.255.255	192.168.2.2
Net mask	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
PRP Listen TCP Port	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.

To create an Ethernet/TCP or Ethernet/UDP peripheral conversation, the IP Address and port are specified in the PRP message body of an **Open** command sent to the **Device** resource. To transfer messages via this peripheral connection the PRP actions **Send** and **Receive** sent to the **Peripheral** resource are used.

After a reset or at power-up, the board retrieves default information for the Prodigy/CME Stand-Alone Ethernet port. To change these default values, the PRP **Set** command is sent to the **Device** resource. See [Section 2.6.3, “Setting Board Defaults,”](#) for a description.

For detailed information on PRP action formats and function, refer to the *Prodigy/CME Programmer's Reference*.

### 2.5.3.1 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 sent 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.

To change the default value of the keep-alive idle time a **Set** command is sent to the **Device** resource. A value of 0 indicates that the keep-alive mechanism should be disabled.

For detailed information on PRP action formats and function, refer to the *Prodigy/CME Programmer's Reference*.

### 2.5.3.2 Connections & associated signals

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

## 2.6 General Board Functions

### 2.6.1 Connection Check

When sent to the **Device** resource, this PRP action **NOP** performs a basic connection check to the Prodigy/CME Stand-Alone board. A return without a PRP error code indicates that the Prodigy/CME Stand-Alone board is accessible and processing commands.

### 2.6.2 Reset

Although a reset occurs automatically during power-up, it is sometimes desirable to reset the Prodigy/CME Stand-Alone board explicitly through a user-initiated command or action. This can be done in one of two ways. The first is via the PRP action **Reset** sent to the **Device** resource. The second method is via the external signal **Reset**, located at pin 91 of the GP Connector.

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 Stand-Alone board's output signals will be driven to known states. These are summarized in the following table:

Signal Name	State
-------------	-------

AxisOutI-4	High
PWMMagIA-4C	Low
PWMSignIA-4B	Low
DACIA-DAC4B	0.0 volts
DigitalOut0-7	Low
AmpEnableI-4	Low
DAC On/Off	Off
Watchdog Timer	Disabled

In addition, upon a board reset all board default parameters are reloaded. See [Section 2.6.3, “Setting Board Defaults,”](#) for more information on default values.

### 2.6.2.1 Connections & associated signals

The reset feature has an external signal input, **Reset**, associated with it. This active low signal is located on pin 91 of the GP Connector. It is pulled up through a 4.7 kOhm resistor to 5 V. In addition, all of the signals in the table above are affected by a board reset.

## 2.6.3 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 power up, 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
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)	Yes
Debug/Console channel	Serial 2

To change the default values used by the board the PRP action **Set** sent to the **Device** resource is used. To read back the current default parameters stored in the board the action **Get** is used. For detailed information on the format of these PRP messages consult the *Prodigy/CME Programmer's Reference*.

## 2.6.4 Dual-ported RAM (Trace Buffer)

The Prodigy/CME Stand-Alone boards have 64 KBytes of on-board dual-ported memory (DPRAM) which is interfaced to the Magellan Motion Control IC as well as directly to the Prodigy/CME Stand-Alone board's communications bus. Typically, the dual-ported RAM is used to capture trace data from the Magellan Motion Control IC. In this mode, its dual port configuration allows for high bandwidth access via the communications bus using PRP actions without affecting Magellan command traffic. For more information on using the Magellan's trace capability, see the *Magellan Motion Control IC User Guide*.

The Magellan Motion Control IC can be programmed to store information into the trace buffer using virtual buffers of various sizes. Thus to correctly read the contents of the dual-ported RAM when used in this mode, you must know the memory organization of the Magellan's trace function. See the *Magellan Motion Control IC User Guide* for details.

Another potential use of this RAM buffer is as a general purpose memory or register-based communications portal. Since the dual-ported RAM is a resource addressable through the PRP system, the originator of the data, as well as the retriever of the data, can be any device on, or external to, the board. While there are several possible ways to utilize the dual-ported RAM in this way, the most common configuration is for the user application code of the C-Motion Engine to collect and store data to the dual-ported RAM buffer, and to have an external host controller read it.

To read or write to the dual-ported RAM using the PRP system, a resource address must first be obtained by sending the **Open** action to the **Memory** resource. This resource address is used for all further access to the RAM. To write data to the dual-ported RAM, the action **Write** is sent to the memory resource. To read the contents of the dual-ported RAM, the action **Read** is used. Note that byte-sized memory operations are not supported to the dual-ported RAM.



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

For complete information on the format and function of these, and other PRP actions, refer to the *Prodigy/CME Programmer's Reference*.

## 2.6.5 Non-volatile Memory

The Prodigy/CME Stand-Alone boards have a general purpose 2,000 (16-bit) word 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 1,999. When writing to this memory, a typical write takes 30  $\mu$ Secs, however under certain circumstances it can take much longer, up to several 100 mSec. Read speed is a few microseconds. 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 is similar to an EEPROM and can be rewritten a limited number of times. The worst case write limit cycle is several 100,000 times. 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.

For complete information on the format and function of these, and other PRP actions, refer to the *PMD Resource Access Protocol Programming Reference*.



The typical write time to the non-volatile RAM is 30  $\mu$ Sec, 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.



## 2.7 Signal Processing and Hardware Functions

These functions are implemented in hardware and are not directly user-programmable. The following sections are organized into related groups of signals, and provide information which may be helpful when connecting the motion system.

### 2.7.1 Home, AxisIn, AxisOut, Limits, Hall Sensors

These signals are conditioned by the board, and then input or output 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.

These signals are named *HomeI-4*, *AxisInI-4*, *AxisOutI-4*, *PosLimI-4* (positive direction limit input), *NegLimI-4* (negative direction limit input), and *HallIA-4C* (12 signals in all).

#### 2.7.1.1 Connections & associated signals

These signals are single-ended digital inputs to the board, with the exception of *AxisOut*, which is a single-ended output. There are no associated connections required for these signals to function properly, however, one or more of the digital grounds must be connected. The input signals are pulled up through 4.7 kOhm resistors to 5V. The default power-up value for all *AxisOut* signals is high.

See [Chapter 4, Electrical Reference](#), for a complete description of the pinout connections to and from the board.

### 2.7.2 QuadA, QuadB, Index

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 microstepping or step (pulse & direction) 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. These signals are named *QuadA1+* through *QuadB4-* (16 signals), and *Index1+* through *Index4-* (8 signals).

#### 2.7.2.1 Connections & associated signals

These 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 + signal is connected, and the - signal should be left floating. For example, in connecting to the A quadrature input, *QuadA1+* connects to the 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 Stand-Alone board. See the *Magellan Motion Control IC User Guide* for more information. Associated connections supported by the board are the +5V output signals. These are provided as a convenience, as they are generally connected to a corresponding input on the encoder to power its internal circuitry. As was the case for the digital input signals, one or more of the digital grounds must also be connected.

See [Chapter 4, Electrical Reference](#), for a complete description of the pinout connections to and from the board.

## 2.7.3 Analog Input

The *Analog I-8* signals provide general purpose input of up to eight analog signals. The voltages present at these various connections do not directly affect the Prodigy/CME Stand-Alone board's behavior. However, they can be read through the Prodigy/CME Stand-Alone board, thus providing a convenient way of importing analog signal levels which may be acted upon by the User application code located on the host PC. These signals are read using the Magellan command **ReadAnalog**. For more information on reading the value of these analog inputs, see the *Magellan Motion Control IC User Guide*. The minimum allowed input voltage is 0.0V, and the maximum allowed input voltage is 3.3V. To determine the numerical value that will be read by the Prodigy/CME Stand-Alone board given a specific voltage, the following formula is used:

$$\text{ReadValue} = \text{AnalogVoltage} * 65,536 / 3.3\text{V}$$

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

$$\text{AnalogVoltage} = \text{ReadValue} * 3.3\text{V} / 65,536$$

### 2.7.3.1 Connections & associated signals

For analog voltages to be read correctly, in addition to the analog signal itself, *AnalogGND* (analog ground) must be connected.

## 2.7.4 Pulse & Direction

For pulse & direction applications, these signals provide a stream of pulse and direction data, and are compatible with a wide variety of off-the-shelf step motor amplifiers. These signals are generated by the Magellan Motion Control IC and are named *Pulse I-4* and *Direction I-4*. The default value at power-up and reset for all pulse and direction output signals is: pulse signal is high; direction signal is low.

### 2.7.4.1 Connections & associated signals

Both single-ended and differential line driver versions of these signals are output from the Prodigy/CME Stand-Alone board. There are no associated connections required for these signals to function properly, however, one or more of the digital grounds must be connected. See [Chapter 4, Electrical Reference](#), for a complete description of the pinout connections to and from the board.

## 2.7.5 PWM Out

For DC brush, brushless DC or microstepping motors, these signals provide PWM (pulse width modulated) motor command signals when the motor output mode is set to **PWMSignMagnitude** or **PWM5050Magnitude**. The number of signals per axis varies, depending on factors such as the motor type, the number of phases of the motor, and the motor drive method (sign/magnitude or 50/50). See [Chapter 4, Electrical Reference](#), for complete connection tables for various motor configurations.

These signals are named *PWMMag IA-4C* (12 signals) and *PWMSign IA-4B* (8 signals).

### 2.7.5.1 Connections & associated signals

These signals are generated by the Magellan Motion Control IC. There are no associated connections required for these signals to function properly, however, one or more of the digital grounds must be connected. See [Chapter 4, Electrical Reference](#), for a complete description of the pinout connections to and from the board.

## 2.7.6 DAC Out

For DC brush, brushless DC or microstepping motors, this is the analog motor command when the motor output mode is set to DAC Offset Binary (digital-to-analog converter). These signals are named **DAC1A - DAC4B** (8 signals), and vary between -10V and +10V. The number of signals per axis depends upon the motor type. See [Section 4.5](#), “[Environmental and Electrical Ratings](#),” and the *Magellan Motion Control IC User Guide* for more information.

### 2.7.6.1 Connections & associated signals

For analog voltages to be output correctly, **AGND** (motor command ground) must be connected. See [Chapter 4, \*Electrical Reference\*](#), for a complete description of the pinout connections to and from the board.

## 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 Stand-Alone boards. In addition, a special library of commands is used to access Magellan Motion Control IC functions called VB-Motion, and C-Motion.

The *Prodigy/CME Programmer's Reference*, in addition to providing detailed format and function descriptions of the PRP messages associated with the Prodigy/CME Stand-Alone 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 Stand-Alone 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 Stand-Alone 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.

For a complete description of the C-language commands supported by the Prodigy/CME Stand-Alone board see the *Prodigy/CME Programmer's Reference*. For a complete description of Magellan C-Motion commands, see the *Magellan Motion Control IC Programming Reference*. For a detailed description of how to develop, download, and monitor user application code for the C-Motion Engine see the *C-Motion Engine Development Tools* manual.

This page intentionally left blank.

# 3. Accessing Board Resources

3

## In This Chapter

- ▶ Resource Addressing
- ▶ Accessing the Communications Ports
- ▶ Accessing On-Board Resources
- ▶ Accessing Magellan-Attached Devices
- ▶ PRP Communication Formats

## 3.1 Resource Addressing

The Prodigy/CME Stand-Alone board's various features and capabilities are organized into groups called 'resources.' For example, to communicate with the Prodigy/CME Stand-Alone board's Magellan Motion Control IC, the host controller sends a command to the **MotionProcessor** resource.

To address different types of resources, the Prodigy/CME Stand-Alone boards provides a general purpose resource addressing scheme called the *PMD Resource access Protocol* (PRP). PRP is a sophisticated general purpose addressing scheme that allows:

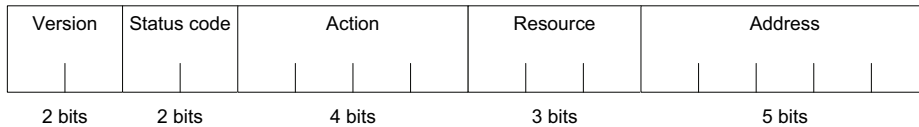
- one or more host controllers to communicate with Prodigy/CME Stand-Alone boards
- Prodigy/CME Stand-Alone boards to communicate with each other
- Prodigy/CME Stand-Alone boards to communicate with other PMD products such as IONs and non-/CME Prodigy boards
- Prodigy/CME Stand-Alone boards to communicate with user-designed hardware, or other off-the-shelf hardware.

### 3.1.1 PMD Resource Access Protocol (PRP)

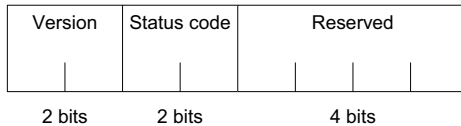
The core of the PMD Resource access Protocol is a header that accompanies all PRP communications. [Figure 3-1](#) shows the format of the resource access protocol header. The PRP header is a single 16-bit word divided into five fields. Normally, the PRP header is immediately followed by a message body, but there are certain communications that do not require a message body. PRP headers are used in both the outgoing, and the response packet. The returned PRP header is 1 byte in length, consisting of the version field, the status code field, and 4 reserved bits.

**Figure 3-1:**  
**Outgoing and**  
**Returning PRP**  
**header formats**

### Outgoing PRP Header



### Return PRP Header



The majority of communications to/from the host controller and the Prodigy/CME Stand-Alone board use the PRP header. Exceptions are 'low-level' communications sent or received directly from the Serial, CANbus, or Ethernet ports. See [Section 3.2, "Accessing the Communications Ports,"](#) for more information.

PRP header field descriptions:

**Version** - This two bit field encodes the version of PRP being used. The value of this field for the Prodigy/CME Stand-Alone board should always be 1 (binary 01) unless documentation included with your Prodigy/CME Stand-Alone board indicates otherwise.

**Status code** - For PRP commands being sent out, this 2-bit field should contain the value 2. When received, a return value of 1 indicates that an error occurred during PRP command processing. If this is the case additional return codes may be stored in the message body. If the return value is 0 then the PRP command was processed successfully.

**Action** - This 4-bit field contains an action identifier that is used to process PRP messages. See [Section 3.1.3, "PRP Actions,"](#) for a summary of the PRP actions supported by the Prodigy/CME Stand-Alone board. This field is not used in the return PRP header.

**Resource** - This 3-bit field encodes the specific resource being addressed. See the table in [Section 3.1.2, "PRP Resources,"](#) for the complete resource map of the Prodigy/CME Stand-Alone board. This field is not used in the return PRP header.

**Address** - This 5-bit field encodes the address of the particular resource being communicated to. Fixed addresses allow on-board resources to be addressed. See the table in [Section 3.1.2, "PRP Resources,"](#) for a resource map of these addresses. Automatically assigned addresses are used to access attached devices, and are also used to create peripheral connections, which are communication 'conversations' between the Prodigy/CME Stand-Alone board and another device. This field is not used in the return PRP header.

The following sections provide general information on the PRP system. For a detailed description of the PRP header, resources, and supported actions, see the *Prodigy/CME Programmer's Reference*.

## 3.1.2 PRP Resources

The Prodigy/CME Stand-Alone board supports five different resource types, each of which is identified by a specific Resource ID. In addition each available resource has an Address. The following table summarizes these Resource IDs and Addresses for the on-board Prodigy/CME Stand-Alone resources:

Resource Name	Resource ID	Address	Comments
Device	0	0	The Device resource indicates a Prodigy/CME Stand-Alone board.

Resource Name	Resource ID	Address	Comments
CMotionEngine	1	0	The CMotionEngine resource indicates a C-Motion Engine.
MotionProcessor	2	0	The MotionProcessor resource indicates a Magellan Motion Control IC.
Memory	3	0	The Memory resource indicates a dual-ported RAM (random access memory).
Peripheral	4	0	The Peripheral resource indicates a communications connection. A peripheral address of 0 indicates a 'null' peripheral. See <a href="#">Section 3.2.1, "Peripheral Connections,"</a> for more information.

### 3.1.3 PRP Actions

The Prodigy/CME Stand-Alone board supports ten different actions, each of which is identified by a specific Action ID. In addition, many actions have sub-actions associated with them that are loaded into the PRP message body. For more information on actions, sub-actions, and the exact format by which the message body should be loaded, see the *Prodigy/CME Programmer's Reference*. The following table summarizes the Action IDs for the Prodigy/CME Stand-Alone boards:

Name	Action ID	Used by Resource	Comments
NOP	0	All	The No Operation (NOP) command is used to verify connection to a given resource.
Reset	1	Device, MotionProcessor	Resets the specified resource.
Command	2	MotionProcessor, CMotionEngine	Sends a command to the specified resource.
Open	3	Peripheral, Device	Creates a new addressable resource. Resource addresses created using the Open action are not fixed, they are assigned automatically at the time a connection is requested.
Close	4	Peripheral, Device, MotionProcessor	Closes a resource created by Open, freeing the automatically-assigned address. Used when a resource is no longer needed.
Send	5	Peripheral, C-Motion Engine	Sends a message to the specified resource.
Receive	6	Peripheral, C-Motion Engine	Receives a message from the specified resource
Write	7	Memory	Writes a data word to a memory resource
Read	8	Memory	Reads a data word from a memory resource
Set	9	Device, CMotionEngine	Sets parameters for a specified resource
Get	10	Device, CMotionEngine	Get parameters for a specified resource

## 3.2 Accessing the Communications Ports

[Figure 2-1](#) shows an internal block diagram of the Prodigy/CME Stand-Alone board. Routing of communications within the board is accomplished via a high-speed communications bus. This bus interconnects resources within the board including the Magellan Motion Control IC, the C-Motion Engine, and the Dual-ported RAM. External communications use this same bus to access the serial, CANbus, or Ethernet communication ports. This means that commands to board resources may originate from outside the board via the serial, CANbus, or Ethernet ports, or internally, via the C-Motion Engine.

A powerful feature of the Prodigy/CME Stand-Alone board's communication bus is that it can process multiple communication requests simultaneously. For example the C-Motion Engine can be used to send motion commands to the on-board Magellan Motion Control IC, while an external host controller can communicate to the same Magellan

via the Ethernet port to monitor progress of the moves, and display results on a remote capture/analysis program such as PMD's Pro-Motion software.



While it is allowed to have more than one communications channel send motion commands to the Prodigy/CME Stand-Alone board's Magellan Motion Control IC and another channel send monitoring-related requests, it is not advisable to have multiple communication channels send motion commands to the Magellan at the same time. This may result in unexpected or unsafe motion.

### 3.2.1 Peripheral Connections

The Prodigy/CME Stand-Alone board provides three different network connection types, Serial, CANbus, and Ethernet. These communication resources are represented in PRP by a construct called a peripheral connection. A peripheral is a resource (resource ID: 4), and is provided, or utilized, by various PRP actions to send and receive messages to network connections.

Obtaining access to either the Serial, CANbus, or Ethernet port is accomplished via the PRP **Open** action. This action opens a peripheral by specifying a sub-action of **OpenSerial**, **OpenCAN**, or **OpenTCP** or **OpenUDP**, as well as the detailed connection parameters that will be used during communications with that specific peripheral connection. Each new open peripheral connection receives an automatically assigned address. The application code that requests the new peripheral connection must record that provided address for future use, and it is this address that is used within the PRP message to reference the newly created peripheral connection.



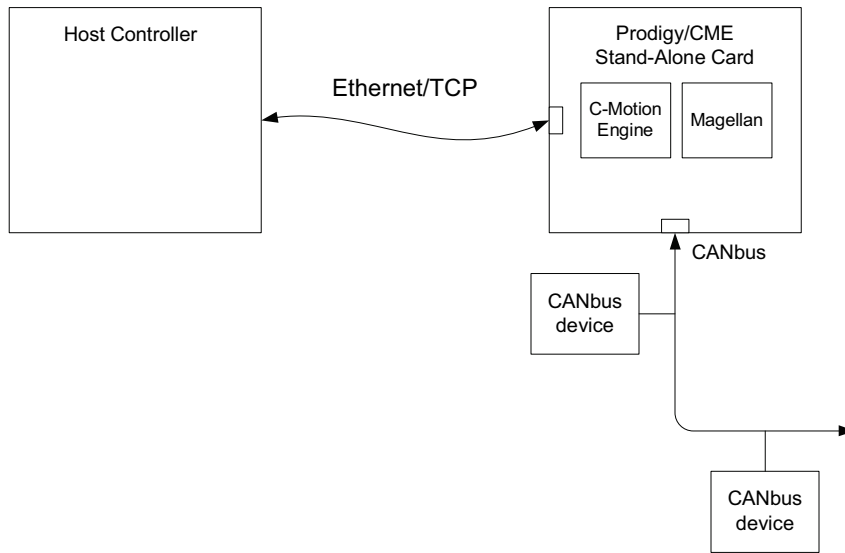
Automatically assigned addresses generally increment by one each time they are assigned, however this should not be assumed. The exact return address value should be stored and only that value should be used to reference that specific peripheral. Each **Peripheral Open** action is specific to the type of connection being requested. For example there is an 'open peripheral' command for CANbus communications, one for Serial communications, and so on.

To send a message to the new peripheral connection, a timeout parameter and the desired message is loaded into the PRP message body and the **Send** action is specified. To retrieve messages from that peripheral connection the **Receive** action is used. The **Receive** action requires that an amount of time (the communication timeout) be loaded into the message body. If a message is not received in the specified amount of time an error is returned.

#### Example

Figure 3-2 shows a network configuration. The host controller needs to initiate, send and receive a message to/from a specific device connected on the CANbus port.





**Figure 3-2:**  
Example  
Network  
Configuration

This sequence will be accomplished in three steps. PRP messages will be assembled that comprise the **Open** action, and then in turn the **Send** and **Receive** actions. As for all PRP messages, the resource ID, the address, and the Action ID must be specified. In this example, to start the sequence the Resource is a 4 specifying a **Peripheral**, the Action ID for **Open** is 2, and the PRP address will be loaded with a 0. The message body is loaded with a code for the sub-action **OpenCANbus** as well as the detailed CANbus communications parameters to establish the connection such as baud rate, send address, and receive address. For the exact message body of this and all PRP actions refer to the *Prodigy/CME Programmer's Reference*.

To simplify the nomenclature for PRP messages, a shorthand nomenclature will be used which contains the mnemonic 'PRP,' the Resource ID, the string 'Addr,' the resource address, and the PRP action. Any additional information pertaining to the error code or message body will be contained in the comment for the message.

```

NewPeriphID = PRP Device, Addr 0, Open    // Send a request to open a connection to the Prodigy/CME
                                           // Stand-Alone board being addressed via the CANbus port.
                                           // The Message body contains the 'CANbus' as well as the
                                           // detailed CANbus parameters for that connection.
                                           // The Address of the new Peripheral connection is contained in
                                           // the body of the return message.

PRP Peripheral, Addr NewPeriphID, Send    // Send a message, contained in the message body, to the CAN
                                           // bus connection created using the Open command. Note that
                                           // the Peripheral address provided in the previous command is
                                           // used to identify which Peripheral connection the message
                                           // should be sent to.

PRP Peripheral, Addr NewPeriphID, Receive // Receive a message. The string will be contained in the message
                                           // body of the return message, from the CANbus Peripheral
                                           // connection.
  
```

Sending a message to the **Peripheral** resource at address 0 has special meaning as the 'null' peripheral. This peripheral address indicates 'no peripheral,' and may be useful in certain situations for disabling console messages, or disabling the Magellan's event output mechanism.



### 3.2.2 Managing Automatically Assigned Addresses

Although most network configurations are created once and left in place while the machine is operating, there may be circumstances where peripheral connections are made, and are then no longer needed. If this is the case, these connections should be ‘closed,’ thereby freeing that automatically assigned **Peripheral** address. This is accomplished via the PRP action **Close**.

The following example illustrates:

A Prodigy/CME Stand-Alone board being used as a production Ethernet board exerciser sends messages to two different Ethernet/TCP addresses and then closes those peripheral connections to allow the process to be repeated indefinitely.

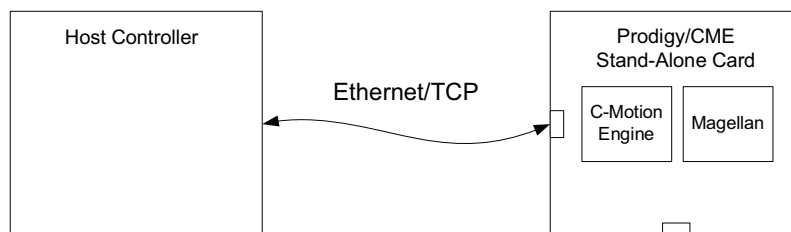
PeriphID1 = PRP Device, Addr 0, Open	// Open a peripheral connection to Ethernet device 1
PRP Peripheral, Addr PeriphID1, Send	// Send a test string to Ethernet device 1
PeriphID2 = PRP Device, Addr 0, Open	// Open a peripheral connection to Ethernet device 2
PRP Peripheral, Addr PeriphID2, Send	// Send a test string to Ethernet device 2
PRP Peripheral, PeriphID2, Close	// Close peripheral connection for Ethernet device 2
PRP Peripheral, PeriphID1, Close	// Close peripheral connection for Ethernet device 1

For complete information on the format and function of these, and other PRP commands, refer to the *Prodigy/CME Programmer's Reference*.

## 3.3 Accessing On-Board Resources

The most common use of the Prodigy/CME Stand-Alone board is to process commands to resources located on the board itself. Typically this means communicating with the on-board Magellan Motion Control IC, but it can also mean communicating with other Prodigy/CME Stand-Alone on-board resources such as the dual-ported RAM. [Figure 3-3](#) shows this configuration.

**Figure 3-3:  
Host Controller  
& On-board  
Resources**



Accessing on-board resources is straightforward using the PRP system. The on-board resource and the address are looked up using the table in [Section 3.1.2, “PRP Resources,”](#) and then a PRP message corresponding to the desired action is sent using those on-board resource and address values. The following example illustrates:

Example 1: A Host Controller wants to set the position of Axis 3 of the Prodigy/CME Stand-Alone’s on-board Magellan Motion Control IC to a value of 0x123456.

From the table in [Section 3.1.2, “PRP Resources,”](#) to communicate with the on-board Magellan Motion Control IC, a PRP message is sent to Resource ID 2 (corresponding to the **MotionProcessor** resource), to address 0 (corresponding to the Prodigy/CME Stand-Alone’s on-board Magellan), with an action ID of 2 (corresponding to the **Command** action). The message body is loaded with the Magellan packet corresponding to “Set Position, #3 0x123456,” which is the 3-word sequence 0x210, 0x0012, 0x3456.

```
PRP MotionProcessor, Addr 0, Command // Send a command to the on-board Magellan Motion Control IC. Mes-
                                     sage
                                     // body contains Magellan Command "SetPosition #3, 0x123456"
```

Upon processing of this command, the host would receive a PRP message back. A zero in the status field would indicate that no error occurred. If this is the case the message body will be empty. If an error did occur, then the PRP status field would contain a 1, and the message body would contain the specific error code that occurred.

Example 2: The Host Controller reads the 32-bit word value of address 0x100 from the Prodigy/CME Stand-Alone's dual-ported RAM

Here is the command that would be sent:

```
PRP Memory, Addr 0, Read // Send a 'Read' action (ID: 7) to the PRP Memory Resource (ID: 3),
                          // address 0 (the address of the Prodigy/CME Stand-Alone on-board
                          // dual-
                          // ported RAM). The PRP message body contains a sub-action of 0,
                          // specifying a 32 bit word read, followed by 0x100 the address of the
                          // 32 bit word to read from the dual-ported RAM.
```

Upon successfully processing this command, the host would receive the 32-bit contents of memory location 0x100 in the message body.

Note that the PRP message to send a Magellan command packet did not use a sub-action code in the message body, while the **Read** command sent to the dual-ported RAM did. Whether or not a sub-action is required, and what the codes are for various sub-actions, is action-specific, and sometimes resource-specific. *The Prodigy/CME Programmer's Reference* provides exact message body information for each PRP action and (if applicable) sub-action.

Note also that in this discussion of sending PRP messages to the Prodigy/CME Stand-Alone board, the specific communications channel to the board (serial, CANbus, Ethernet) was not discussed. That is because the PRP header and message body are the same regardless of how transmission occurs. Depending on the network type used, those varying communication links will send, receive, and process errors for packet communications in different ways, but from the 'perspective' of the PRP system, those message transmission details are handled automatically. This is a very powerful characteristic of the PRP system, and we will see additional examples of similar 'access virtualization' as we discuss more advanced network topologies.

To actually send a PRP command over a serial, CANbus, or Ethernet network, specific protocols must be observed. See [Section 3.5, "PRP Communication Formats,"](#) for this network-specific (serial, CANbus, or Ethernet) information.



## 3.4 Accessing Magellan-Attached Devices

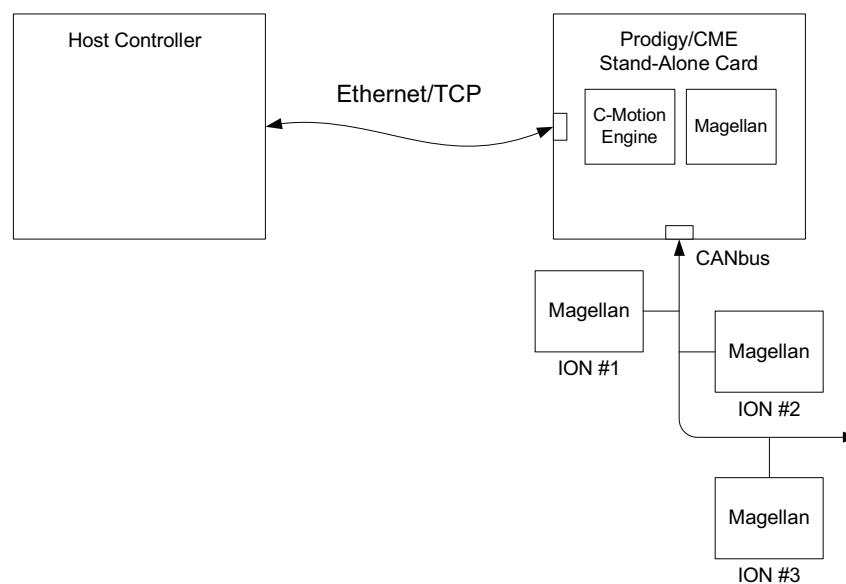
[Section 3.2.1, "Peripheral Connections,"](#) provided information on how general purpose messages can be sent to, or received from, any of the Prodigy/CME Stand-Alone board's network ports. This level of low-level access can be useful to communicate with a wide variety of custom-created or off-the-shelf products that are capable of communicating on that bus.



If the attached device is a PMD device however, such as an ION or non-/CME Prodigy Stand-Alone board (PR825xx20 family and PR925xx20 family) then it is possible to integrate these devices into the PRP access network so that they can be communicated to without using low-level peripheral send and receive commands.

IONs and non-/CME Prodigy Stand-Alone boards are referred to as ‘Magellan-Attached’ devices because the network directly connects to the Magellan Motion Control IC, and utilizes the Magellan’s own communication protocols for receiving commands and returning data along the communication link.

Figure 3-4 provides an example of a setup where a Prodigy/CME Stand-Alone board is connected via Ethernet to a host controller, and where a second network of CANbus-connected IONs is attached to the Prodigy/CME Stand-Alone board’s CANbus port.



In this ‘bridge’ configuration, the host controller (or the Prodigy/CME Stand-Alone board’s C-Motion Engine) can access the CANbus-connected IONs very similarly to the way in which the on-board Magellan is addressed. To accomplish this however, a different method is used to create access to the Magellan compared to the method described for on-board Magellan access in (see [Section 3.3, “Accessing On-Board Resources,”](#) for details).

Rather than sending PRP messages to the on-board **MotionProcessor** resource, first a raw peripheral connection is opened, and then the **Open** action with a sub-action of **MotionProcessor** is used to open a new **MotionProcessor** resource. The following PRP code sequence illustrates:

```

NewPeriphID = PRP Device, Addr 0, Open      // Open a CANbus peripheral connection to connect to ION #1
NewMtnProcID = PRP Peripheral, Addr NewPeriphID, Open
                                              // Use the opened peripheral connection to create a new
                                              // MotionProcessor resource using the Open action with
                                              // sub-action specifying MotionProcessor. A new MotionProcessor
                                              // resource address is loaded into the message body.

PRP MotionProcessor, Addr NewMtnProcID, Command
                                              // Send Magellan command packet to ION #1's Magellan Motion
                                              // Processor using the standard method of communicating with
                                              // MotionProcessor resources.
  
```

The action sequence to communicate with ION #2 and subsequent devices is similar, except the unique CANbus parameters to address that particular ION should be used, and the unique automatically assigned PeriphID and MagellanID values should be recorded and used for resources addressing.

Note that as the command sequence above shows, after the new **MotionProcessor** resource addresses are created using the **Open** action, subsequent Magellan commands to the CANbus IONs are identical in format to commands to the on-board Magellan. This illustrates a very powerful feature of the PRP system which is that it allows resources to be addressed transparently by the host controller (or C-Motion Engine module), making it easy to create and access networks of PMD products.

Note also that the **Open** action can be used with different resource types. In [Section 3.2.1, “Peripheral Connections,”](#) it was used with a resource type of **Device** to open a peripheral connection. In the above example it was used with a resource type of **Peripheral** to open a connection to a Magellan Motion Control IC.

## 3.5 PRP Communication Formats

The following sections discuss how PRP messages should be formatted on the Serial, CANbus, and Ethernet networks so that they can be properly interpreted by the Prodigy/CME Stand-Alone board.

### 3.5.1 PRP Messages over Serial

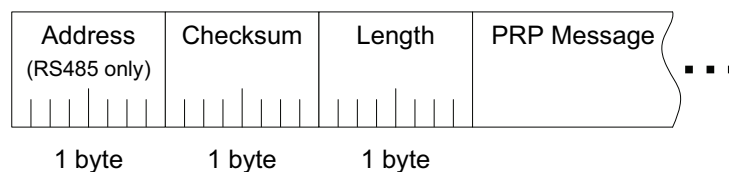
By default, the Prodigy/CME Stand-Alone board is set up to receive PRP command messages from a host controller on Serial port 1. This section describes the format of these packets, which transfer PRP messages over a serial protocol.

[Figure 3-5](#) shows the Serial packet protocol that must be used for all PRP messages. This serial header is 32 bits, and is broken into three fields as follows:

**Address:** the first byte is an address field, used only with RS485 communications. For RS232 communications, this byte is not included in the packet.

**Checksum:** The second byte is a simple 8-bit sum checksum of all bytes in the packet payload (excluding the serial header, i.e., the address, checksum, and length fields). From [Figure 3-5](#) this means only the PRP header and message body contribute to this checksum value

**Length:** The third byte is an 8 bit length field. The length indicates the number of bytes in the packet payload. The packet payload is defined as everything in the serial packet after the length field. That is, the 2 byte PRP header plus however many bytes are contained in the PRP message body. For example if the PRP message body is 6 bytes in size, the value filled into the length field should be 8 (2 bytes for PRP header + 6 bytes for the PRP message body).



**Figure 3-5:**  
PRP Header  
over Serial  
Format

Return PRP messages should be formatted in the same way.

#### 3.5.1.1 Serial packet processing

An error-free Serial/PRP communication sequence from the host controller to the Prodigy/CME Stand-Alone board consists of a full outgoing packet transmission with the correct checksum and specified number of bytes received by the Prodigy/CME Stand-Alone board, and a full packet response with correct checksum and length received at the

host controller. The return message must be received within a fixed amount of time determined by the host controller. Correctly setting this 'timeout window' may depend on factors such as baud rate, but 100 mSec is a typical safe value.

If the host controller receives a response packet with an incorrect checksum, or does not receive the complete response (communications timeout), then the original message should be resent.

If the Prodigy/CME Stand-Alone board receives a packet with an incorrect checksum, then a packet with an error code indicating this is returned to the host controller.

If the Prodigy/CME Stand-Alone board does not receive the specified number of bytes within 100 mSec (the Prodigy/CME Stand-Alone timeout value) of beginning of packet reception, the incoming message is ignored, and no message is sent to the host controller.

### 3.5.1.2 Private versus PRP serial message use

For serial port traffic that is strictly utilized by the user application code, raw messages can be sent and received using the **Peripheral** mechanism (see [Section 3.2.1, "Peripheral Connections,"](#) for details). The format of the actual message is created and interpreted by the User application code, and there is no need to implement the PRP serial header (see [Section 3.2.1, "Peripheral Connections,"](#) for details).

For serial port traffic that is intended to communicate with the User application code as well as Prodigy/CME Stand-Alone on-board resources such as the Magellan Motion Control IC or the C-Motion Engine, a PRP-compatible user packet must be used.

#### Example

A PC host will communicate by serial port to a Prodigy/CME Stand-Alone board which is functioning as a 3 axis robot controller. The user application loaded on the C-Motion Engine implements a custom command set to control the robot. In addition, the PC host will send commands to the on-board Magellan Motion Control IC for diagnostics and robot setup. In this instance, the custom command set used to control the robot would need to be built within the PRP system so that Magellan and other Prodigy/CME Stand-Alone resource traffic is properly separated from the user's application traffic. This simplest way to do this would be to use the built-in C-Motion Engine PRP user packet **Send** and **Receive** capability (see [Section 2.4.4, "Sending Messages to/from User Application Code,"](#) for details).

## 3.5.2 PRP Messages over CANbus

If the Prodigy/CME Stand-Alone board is set up to process PRP command messages from a host controller over CANbus, a specific format for the packets must be followed. This section describes the format of how PRP messages are carried over CANbus.

Since native CANbus communications can not be larger than 8 bytes, hosting the PRP system, which can support shorter as well as longer messages, requires additional layers to manage data segmentation and desegmentation. The protocol that is used by the Prodigy/CME Stand-Alone boards to accomplish this is very similar to the SDO (Service Data Object) protocol of the CANopen standard.

The details of this protocol are extensive enough that they are not described here, but are available in the *Prodigy/CME Programmer's Reference*.

### 3.5.2.1 CANbus packet processing

Unlike the serial protocols, the SDO based CANbus protocol has a robust error checking and retransmission mechanism built in that corrects for garbled or otherwise unusable transmissions.

Nevertheless, if a host controller does not receive the complete response packet within a specific time window (communications timeout), then the original message should be resent.

### 3.5.2.2 Private versus PRP CANbus message use

A long as the addresses utilized for user application code are not specified the same as the PRP send/receive and event port, there are no limitations to intermixing PRP and user application messages over a CANbus network.

## 3.5.3 PRP Messages over Ethernet

The existence of ports, and the broad range of packet lengths that are supported with the Ethernet protocol, makes sending PRP messages very simple. For both sent and received messages the PRP message is simply loaded as the ‘payload’ of the Ethernet message. The only convention that must be observed is that the host controller’s destination TCP port must be equal to the Prodigy/CME Stand-Alone board’s TCP default value set using the **SetDefault** command. Note that the UDP protocol may not be used for PRP communications to/from the Prodigy/CME Stand-Alone board.

### 3.5.3.1 Ethernet packet processing

Unlike the serial protocols, Ethernet TCP packets have a robust error checking and retransmission mechanism built in that corrects for garbled or otherwise unusable transmissions.

### 3.5.3.2 Private versus PRP Ethernet message use

A long as the ports utilized for user application code are not specified the same as the PRP send/receive and event port, there are no limitations to intermixing PRP and user application messages to the device at a specific IP address.

User application code can open either Ethernet/TCP or Ethernet/UDP peripherals, but there are some important differences in how the Ethernet protocol processes them. The most important difference is that UDP is a connectionless protocol, and data is not guaranteed to be received in the order they were sent, nor are they guaranteed to be received at all. If a UDP packet is lost it will not be retransmitted.

UDP is thus best suited for applications in which low latency is more important than guaranteed delivery, such as data logging, or in applications where there are other methods by which data transmission can be confirmed.

This page intentionally left blank.



# 4. Electrical Reference

## In This Chapter

- ▶ User-Settable Components
- ▶ Connectors
- ▶ Connections Summary—Motor Amplifiers
- ▶ Cables
- ▶ Environmental and Electrical Ratings
- ▶ Certifications & Compliance
- ▶ Mechanical Dimensions
- ▶ User I/O memory Map

## 4.1 User-Settable Components

Figure 4-1 illustrate the locations of the principal components of the Prodigy/CME Stand-Alone boards. The user-settable components of the board are listed in the following table:

Component	Function
Resistor packs RS1, RS2, and RS3	Sets the encoder termination.
SW1 DIP switch	Sets the GP Connector Motor Output Configuration

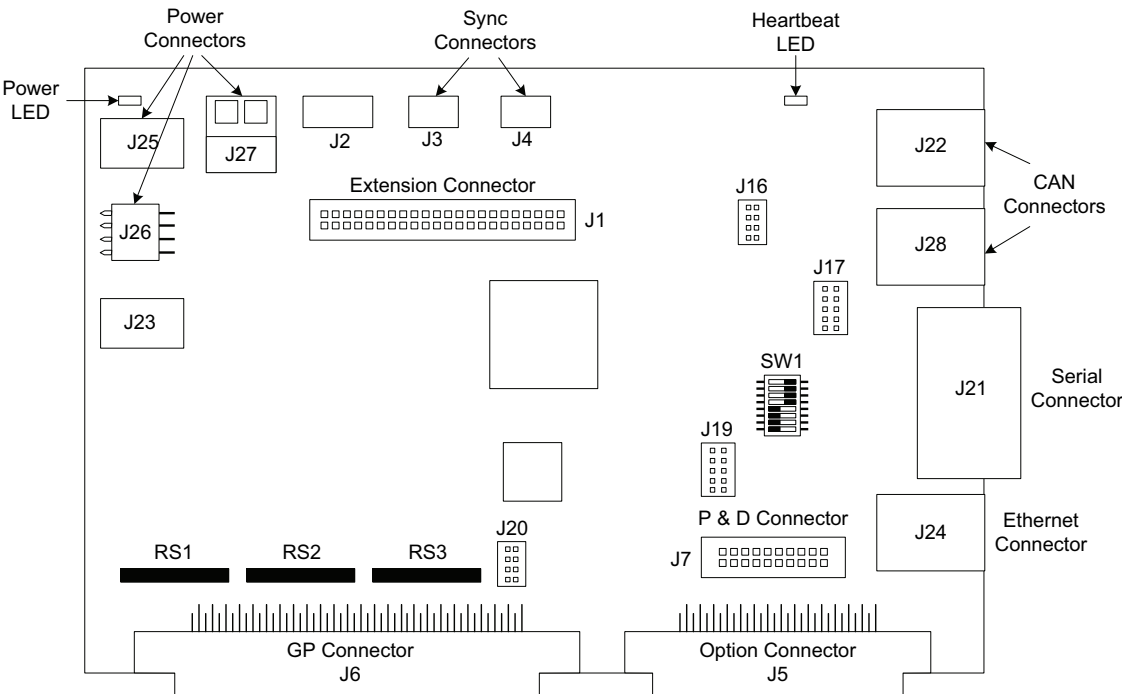


Figure 4-1:  
Components  
and layout,  
front of board

### 4.1.1 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 employed, resistor packs RS1, RS2, and RS3 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 RS1, RS2, RS3	Installed; this is the default setting of resistor packs RS1 - RS3.	If differential connections are being used, leave the resistor packs installed.
	Removed	If single-ended encoder connections are being used, remove the resistor packs.

#### 4.1.1.1 Differential Encoder Connections

Differential encoder connections are detailed in the following table. All connections are made through the J6 GP Connector.:

Signal	J6 (GP Connector) Pin Connections			
	Axis 1	Axis 2	Axis 3	Axis 4
QuadAn+	J6-1	J6-37	J6-2	J6-38
QuadAn-	J6-3	J6-39	J6-4	J6-40
QuadBn+	J6-5	J6-41	J6-6	J6-42
QuadBn-	J6-7	J6-43	J6-8	J6-44
Indexn+	J6-9	J6-45	J6-10	J6-46
Indexn-	J6-11	J6-47	J6-12	J6-48
Vcc	J6-13	J6-49	J6-14	J6-50
GND	J6-15	J6-51	J6-16	J6-52

PMD's Cable-5003-01.R provides a bifurcated 100 pin to dual 50-pin header cable. For your convenience, the following table provides encoder connections numbered for these cable headers rather than the GP Connector. "H1" refers to Header 1, and "H2" refers to Header 2 of this cable.

Signal	Cable-5003-01.R Connections			
	Axis 1	Axis 2	Axis 3	Axis 4
QuadAn+	H1-1	H1-19	H2-1	H2-19
QuadAn-	H1-2	H1-20	H2-2	H2-20
QuadBn+	H1-3	H1-21	H2-3	H2-21
QuadBn-	H1-4	H1-22	H2-4	H2-22
Indexn+	H1-5	H1-23	H2-5	H2-23
Indexn-	H1-6	H1-24	H2-6	H2-24
Vcc	H1-7	H1-25	H2-7	H2-25
GND	H1-8	H1-26	H2-8	H2-26

#### 4.1.1.2 Single-Ended Encoder connections

Single-ended encoder connections are detailed in the following table. All connections are made through the J6 GP connector.

Signal	J6 (GP Connector) Pin Connections			
	Axis 1	Axis 2	Axis 3	Axis 4
QuadAn	J6-1	J6-37	J6-2	J6-38
QuadBn	J6-5	J6-41	J6-6	J6-42
Indexn	J6-9	J6-45	J6-10	J6-46
Vcc	J6-13	J6-49	J6-14	J6-50
GND	J6-15	J6-51	J6-16	J6-52

PMD's Cable-5003-01.R provides a bifurcated 100 pin to dual 50-pin header cable. For your convenience, the following table provides encoder connections numbered for these cable headers rather than the GP Connector. "H1" refers to Header 1, and "H2" refers to Header 2 of this cable.

Signal	Cable-5003-01.R Connections			
	Axis 1	Axis 2	Axis 3	Axis 4
QuadAn+	H1-1	H1-19	H2-1	H2-19
QuadBn+	H1-3	H1-21	H2-3	H2-21
Indexn+	H1-5	H1-23	H2-5	H2-23
Vcc	H1-7	H1-25	H2-7	H2-25
GND	H1-8	H1-26	H2-8	H2-26

#### 4.1.1.3 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.

### 4.1.2 Motor Output Configuration

There are two configurations for setting up the motor output pins on the Prodigy/CME Stand-Alone GP Connector: These motor output pins can be set up for use with servo motors (DC Brush or Brushless DC) or step (pulse & direction) motors. The default setting is for all motors to be set for servo motors.

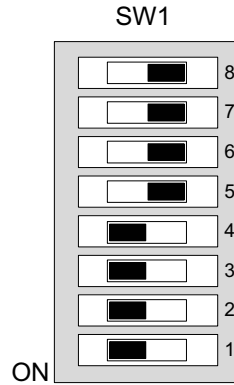
The switch bank SW1 controls these settings, which apply on a per-axis basis. The following table describes the switch setting for each output type. [Figure 4-2](#) provides a diagram of SW1. See [Figure 4-1](#) for SW1 location on the Prodigy/CME Stand-Alone board. In the following table, the individual switches are numbered from left to right.

Axis	SW1 Set for Servo Motor	SW1 Set for Pulse & Direction
1	1 on, 5 off	1 off, 5 on
2	2 on, 6 off	2 off, 6 on
3	3 on, 7 off	3 off, 7 on
4	4 on, 8 off	4 off, 8 on

Power to the board should be turned off when changing the state of the SW1 switches, and care should be taken not to set both switch pairs on (for example both 1 and 5 on, or both 2 and 6 on etc.) or damage may occur to the board.



**Figure 4-2:**  
Switch  
Settings by  
Output Type



## 4.2 Connectors

There are 15 user-accessible connectors on the Prodigy/CME Stand-Alone board. See [Figure 4-1](#) 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
GP	J6	Provides various motion connections to external amplifiers and motor elements.
Option	J5	Used with brushless DC and microstepping motors, the Option Connector connector provides additional connections to external amplifier and motor components.
Serial	J21	Dual serial port for serial RS232 or RS485 connections.
Serial Alt.	J17	Electrical duplicate of the Serial Connector, but has an alternate physical connector type. It may be useful for certain board mounting configurations.
Sync I/O (x2)	J3, J4	Synchronizes additional boards within the same motion system.
CAN1, CAN2	J22, J28	Provides connection to a CAN 2.0B network.
Ethernet	J24	Provides connection to an Ethernet TCP/IP network
Pulse & Direction	J7	Provides pulse & direction differential output.
Power	J26	Provides power connections to the board suitable for cable harnesses.
Power Alt1,	J27	Provides power to the board with identical electrical connections as the Power Connector, suitable for field termination.
Power Alt2*	J25	Provides power to the board with identical electrical connections as the Power Connector, suitable for connection to a plug-in wall outlet power supply.

\*Power Alt2 is only populated for the Developer's Kit version of the Prodigy/CME Stand-Alone board.

### 4.2.1 GP Connector

The GP Connector (J6 in [Figure 4-1](#)) connects to various motion peripherals such as encoders, amplifiers, etc. The GP Connector is comprised of a single 100-pin high density connector, which if connected to with PMD p/n Cable-5003-01.R splits (bifurcates) into two 50-pin ribbon cables with headers. See [Section 4.4.2, “Cable-5003-01.R,”](#) for detailed information on this cable.

## 4.2.2 GP Connector Using DC Brush Motors

Pin	Connection	Description	Pin	Connection	Description
<b>J6</b>					
1	QuadA1+	Quadrature A+ encoder input (axis 1)	2	QuadA3+	Quadrature A+ encoder input (axis 3)
3	QuadA1-	Quadrature A- encoder input (axis 1)	4	QuadA3-	Quadrature A- encoder input (axis 3)
5	QuadB1+	Quadrature B+ encoder input (axis 1)	6	QuadB3+	Quadrature B+ encoder input (axis 3)
7	QuadB1-	Quadrature B- encoder input (axis 1)	8	QuadB3-	Quadrature B- encoder input (axis 3)
9	Index1+	Index+ input (axis 1)	10	Index3+	Index+ input (axis 3)
11	Index1-	Index- input (axis 1)	12	Index3-	Index- input (axis 3)
13	Vcc	+5V	14	Vcc	+5V
15	GND	Ground	16	GND	Ground
17	PosLim1	Pos. direction limit switch input (axis 1)	18	PosLim3	Pos. direction limit switch input (axis 3)
19	NegLim1	Neg. direction limit switch input (axis 1)	20	NegLim3	Neg. direction limit switch input (axis 3)
21	Home1	Home input (axis 1)	22	Home3	Home input (axis 3)
23	GND	Ground	24	GND	Ground
25	AxisOut1	AxisOut output (axis 1)	26	AxisOut3	AxisOut output (axis 3)
27	PWMMag1A	PWM magnitude output (axis 1)	28	PWMMag3A	PWM magnitude output (axis 3)
29	PWMSign1A	PWM sign output (axis 1)	30	PWMSign3A	PWM sign output (axis 3)
31	AxisIn1	AxisIn input (axis 1)	32	AxisIn3	AxisIn input (axis 3)
33	DAC1A	Analog mtr cmd output (axis 1), $\pm 10V$	34	DAC3A	Analog mtr cmd output (axis 3), $\pm 10V$
35	AGND	Ground for analog motor command	36	AGND	Ground for analog motor command
37	QuadA2+	Quadrature A+ encoder input (axis 2)	38	QuadA4+	Quadrature A+ encoder input (axis 4)
39	QuadA2-	Quadrature A- encoder input (axis 2)	40	QuadA4-	Quadrature A- encoder input (axis 4)
41	QuadB2+	Quadrature B+ encoder input (axis 2)	42	QuadB4+	Quadrature B+ encoder input (axis 4)
43	QuadB2-	Quadrature B- encoder input (axis 2)	44	QuadB4-	Quadrature B- encoder input (axis 4)
45	Index2+	Index+ input (axis 2)	46	Index4+	Index+ input (axis 4)
47	Index2-	Index- input (axis 2)	48	Index4-	Index- input (axis 4)
49	Vcc	+5V	50	Vcc	+5V
51	GND	Ground	52	GND	Ground
53	PosLim2	Pos. direction limit switch input (axis 2)	54	PosLim4	Pos. direction limit switch input (axis 4)
55	NegLim2	Neg. direction limit switch input (axis 2)	56	NegLim4	Neg. direction limit switch input (axis 4)
57	Home2	Home input (axis 2)	58	Home4	Home input (axis 4)
59	AxisOut2	AxisOut output (axis 2)	60	AxisOut4	AxisOut output (axis 4)
61	PWMMag2A	PWM magnitude output (axis 2)	62	PWMMag4A	PWM magnitude output (axis 4)
63	PWMSign2A	PWM sign output (axis 2)	64	PWMSign4A	PWM sign output (axis 4)
65	AxisIn2	AxisIn input (axis 2)	66	AxisIn4	AxisIn input (axis 4)
67	DAC2A	Analog mtr cmd output (axis 2), $\pm 10V$	68	DAC4A	Analog mtr cmd output (axis 4), $\pm 10V$
69	AGND	Ground for analog motor command	70	AGND	Ground for analog motor command
71	DigitalIn0	General purpose digital input 0	72	DigitalIn4	General purpose digital input 4
73	DigitalIn1	General purpose digital input 1	74	DigitalIn5	General purpose digital input 5
75	DigitalIn2	General purpose digital input 2	76	DigitalIn6	General purpose digital input 6
77	DigitalIn3	General purpose digital input 3	78	DigitalIn7	General purpose digital input 7
79	AmpEnable1	Amplifier enable signal (axis 1)	80	AmpEnable3	Amplifier enable signal (axis 3)
81	DigitalOut0	General purpose digital output 0	82	DigitalOut4	General purpose digital output 4
83	DigitalOut1	General purpose digital output 1	84	DigitalOut5	General purpose digital output 5
85	DigitalOut2	General purpose digital output 2	86	DigitalOut6	General purpose digital output 6
87	DigitalOut3	General purpose digital output 3	88	DigitalOut7	General purpose digital output 7
89	AmpEnable2	Amplifier enable signal (axis 2)	90	AmpEnable4	Amplifier enable signal (axis 4)
91	Reset	Hardware reset input	92	AnalogGND	Gnd for general purpose analog inputs
93	Analog1	General purpose analog input 1	94	Analog5	General purpose analog input 5
95	Analog2	General purpose analog input 2	96	Analog6	General purpose analog input 6
97	Analog3	General purpose analog input 3	98	Analog7	General purpose analog input 7
99	Analog4	General purpose analog input 4	100	Analog8	General purpose analog input 8

Note: If used, Cable 5003-01.R will map signals on J6 to different pin numbers as compared to this table. See [Section 4.4.2, “Cable-5003-01.R,”](#) for details.

### 4.2.3 GP Connector Using Brushless DC or Microstepping Motors

Pin	Connection	Description	Pin	Connection	Description
<b>J6</b>					
1	QuadA1+	Quadrature A+ encoder input (axis 1)	2	QuadA3+	Quadrature A+ encoder input (axis 3)
3	QuadA1-	Quadrature A- encoder input (axis 1)	4	QuadA3-	Quadrature A- encoder input (axis 3)
5	QuadB1+	Quadrature B+ encoder input (axis 1)	6	QuadB3+	Quadrature B+ encoder input (axis 3)
7	QuadB1-	Quadrature B- encoder input (axis 1)	8	QuadB3-	Quadrature B- encoder input (axis 3)
9	Index1+	Index+ input (axis 1)	10	Index3+	Index+ input (axis 3)
11	Index1-	Index- input (axis 1)	12	Index3-	Index- input (axis 3)
13	Vcc	+5V	14	Vcc	+5V
15	GND	Ground	16	GND	Ground
17	PosLim1	Pos. direction limit switch input (axis 1)	18	PosLim3	Pos. direction limit switch input (axis 3)
19	NegLim1	Neg. direction limit switch input (axis 1)	20	NegLim3	Neg. direction limit switch input (axis 3)
21	Home1	Home input (axis 1)	22	Home3	Home input (axis 3)
23	GND	Ground	24	GND	Ground
25	AxisOut1	AxisOut output (axis 1)	26	AxisOut3	AxisOut output (axis 3)
27	n.c.	No connection	28	n.c.	No connection
29	n.c.	No connection	30	n.c.	No connection
31	AxisIn1	AxisIn input (axis 1)	32	AxisIn3	AxisIn input (axis 3)
33	DAC1A	Phase A analog mtr cmd output (axis 1), $\pm 10V$	34	DAC3A	Phase A analog mtr cmd output (axis 3), $\pm 10V$
35	AGND	Ground for analog motor command	36	AGND	Ground for analog motor command
37	QuadA2+	Quadrature A+ encoder input (axis 2)	38	QuadA4+	Quadrature A+ encoder input (axis 4)
39	QuadA2-	Quadrature A- encoder input (axis 2)	40	QuadA4-	Quadrature A- encoder input (axis 4)
41	QuadB2+	Quadrature B+ encoder input (axis 2)	42	QuadB4+	Quadrature B+ encoder input (axis 4)
43	QuadB2-	Quadrature B- encoder input (axis 2)	44	QuadB4-	Quadrature B- encoder input (axis 4)
45	Index2+	Index+ input (axis 2)	46	Index4+	Index+ input (axis 4)
47	Index2-	Index- input (axis 2)	48	Index4-	Index- input (axis 4)
49	Vcc	+5V	50	Vcc	+5V
51	GND	Ground	52	GND	Ground
53	PosLim2	Pos. direction limit switch input (axis 2)	54	PosLim4	Pos. direction limit switch input (axis 4)
55	NegLim2	Neg. direction limit switch input (axis 2)	56	NegLim4	Neg. direction limit switch input (axis 4)
57	Home2	Home input (axis 2)	58	Home4	Home input (axis 4)
59	AxisOut2	AxisOut output (axis 2)	60	AxisOut4	AxisOut output (axis 4)
61	n.c.	No connection	62	n.c.	No connection
63	n.c.	No connection	64	n.c.	No connection
65	AxisIn2	AxisIn input (axis 2)	66	AxisIn4	AxisIn input (axis 4)
67	DAC2A	Phase A analog mtr cmd output (axis 2), $\pm 10V$	68	DAC4A	Phase A analog mtr cmd output (axis 4), $\pm 10V$
69	AGND	Ground for analog motor command	70	AGND	Ground for analog motor command
71	DigitalIn0	General purpose digital input 0	72	DigitalIn4	General purpose digital input 4
73	DigitalIn1	General purpose digital input 1	74	DigitalIn5	General purpose digital input 5
75	DigitalIn2	General purpose digital input 2	76	DigitalIn6	General purpose digital input 6
77	DigitalIn3	General purpose digital input 3	78	DigitalIn7	General purpose digital input 7
79	AmpEnable1	Amplifier enable signal (axis 1)	80	AmpEnable3	Amplifier enable signal (axis 3)
81	DigitalOut0	General purpose digital output 0	82	DigitalOut4	General purpose digital output 4
83	DigitalOut1	General purpose digital output 1	84	DigitalOut5	General purpose digital output 5
85	DigitalOut2	General purpose digital output 2	86	DigitalOut6	General purpose digital output 6
87	DigitalOut3	General purpose digital output 3	88	DigitalOut7	General purpose digital output 7
89	AmpEnable2	Amplifier enable signal (axis 2)	90	AmpEnable4	Amplifier enable signal (axis 4)
91	Reset	Hardware reset input	92	AnalogGND	Ground for general purpose analog inputs
93	Analog1	General purpose analog input 1	94	Analog5	General purpose analog input 5
95	Analog2	General purpose analog input 2	96	Analog6	General purpose analog input 6
97	Analog3	General purpose analog input 3	98	Analog7	General purpose analog input 7
99	Analog4	General purpose analog input 4	100	Analog8	General purpose analog input 8

Note: If used, Cable 5003-01.R will map signals on J6 to different pin numbers as compared to this table. See Section 4.4.2, "Cable-5003-01.R," for details.

## 4.2.4 GP Connector Using Step (Pulse & Direction) Motors

Pin	Connection	Description	Pin	Connection	Description
<b>J6</b>					
1	QuadA1+	Quadrature A+ encoder input (axis 1)	2	QuadA3+	Quadrature A+ encoder input (axis 3)
3	QuadA1-	Quadrature A- encoder input (axis 1)	4	QuadA3-	Quadrature A- encoder input (axis 3)
5	QuadB1+	Quadrature B+ encoder input (axis 1)	6	QuadB3+	Quadrature B+ encoder input (axis 3)
7	QuadB1-	Quadrature B- encoder input (axis 1)	8	QuadB3-	Quadrature B- encoder input (axis 3)
9	Index1+	Index+ input (axis 1)	10	Index3+	Index+ input (axis 3)
11	Index1-	Index- input (axis 1)	12	Index3-	Index- input (axis 3)
13	Vcc	+5V	14	Vcc	+5V
15	GND	Ground	16	GND	Ground
17	PosLim1	Pos. direction limit switch input (axis 1)	18	PosLim3	Pos. direction limit switch input (axis 3)
19	NegLim1	Neg. direction limit switch input (axis 1)	20	NegLim3	Neg. direction limit switch input (axis 3)
21	Home1	Home input (axis 1)	22	Home3	Home input (axis 3)
23	GND	Ground	24	GND	Ground
25	AxisOut1	AxisOut output (axis 1)	26	AxisOut3	AxisOut output (axis 3)
27	Pulse1	Pulse output (axis 1)	28	Pulse3	Pulse output (axis 3)
29	Direction1	Direction output (axis 1)	30	Direction3	Direction output (axis 3)
31	AxisIn1	AxisIn input (axis 1)	32	AxisIn3	AxisIn input (axis 3)
33	AtRest1	Atrest indicator output (axis 1)	34	AtRest3	Atrest indicator output (axis 3)
35	GND	Ground	36	GND	Ground
37	QuadA2+	Quadrature A+ encoder input (axis 2)	38	QuadA4+	Quadrature A+ encoder input (axis 4)
39	QuadA2-	Quadrature A- encoder input (axis 2)	40	QuadA4-	Quadrature A- encoder input (axis 4)
41	QuadB2+	Quadrature B+ encoder input (axis 2)	42	QuadB4+	Quadrature B+ encoder input (axis 4)
43	QuadB2-	Quadrature B- encoder input (axis 2)	44	QuadB4-	Quadrature B- encoder input (axis 4)
45	Index2+	Index+ input (axis 2)	46	Index4+	Index+ input (axis 4)
47	Index2-	Index- input (axis 2)	48	Index4-	Index- input (axis 4)
49	Vcc	+5V	50	Vcc	+5V
51	GND	Ground	52	GND	Ground
53	PosLim2	Pos. direction limit switch input (axis 2)	54	PosLim4	Pos. direction limit switch input (axis 4)
55	NegLim2	Neg. direction limit switch input (axis 2)	56	NegLim4	Neg. direction limit switch input (axis 4)
57	Home2	Home input (axis 2)	58	Home4	Home input (axis 4)
59	AxisOut2	AxisOut output (axis 2)	60	AxisOut4	AxisOut output (axis 4)
61	Pulse2	Pulse output (axis 2)	62	Pulse4	Pulse output (axis 4)
63	Direction2	Direction output (axis 2)	64	Direction4	Direction output (axis 4)
65	AxisIn2	AxisIn input (axis 2)	66	AxisIn4	AxisIn input (axis 4)
67	AtRest2	Atrest indicator output (axis 2)	68	AtRest4	Atrest indicator output (axis 4)
69	GND	Ground	70	GND	Ground
71	DigitalIn0	General purpose digital input 0	72	DigitalIn4	General purpose digital input 4
73	DigitalIn1	General purpose digital input 1	74	DigitalIn5	General purpose digital input 5
75	DigitalIn2	General purpose digital input 2	76	DigitalIn6	General purpose digital input 6
77	DigitalIn3	General purpose digital input 3	78	DigitalIn7	General purpose digital input 7
79	AmpEnable1	Amplifier enable signal (axis 1)	80	AmpEnable3	Amplifier enable signal (axis 3)
81	DigitalOut0	General purpose digital output 0	82	DigitalOut4	General purpose digital output 4
83	DigitalOut1	General purpose digital output 1	84	DigitalOut5	General purpose digital output 5
85	DigitalOut2	General purpose digital output 2	86	DigitalOut6	General purpose digital output 6
87	DigitalOut3	General purpose digital output 3	88	DigitalOut7	General purpose digital output 7
89	AmpEnable2	Amplifier enable signal (axis 2)	90	AmpEnable4	Amplifier enable signal (axis 4)
91	Reset	Hardware reset input	92	AnalogGND	Ground for general purpose analog inputs
93	Analog1	General purpose analog input 1	94	Analog5	General purpose analog input 5
95	Analog2	General purpose analog input 2	96	Analog6	General purpose analog input 6
97	Analog3	General purpose analog input 3	98	Analog7	General purpose analog input 7
99	Analog4	General purpose analog input 4	100	Analog8	General purpose analog input 8

Note: If used, Cable 5003-01.R will map signals on J6 to different pin numbers as compared to this table. See [Section 4.4.2, "Cable-5003-01.R,"](#) for details.

## 4.2.5 Option Connector

When the Prodigy/CME Stand-Alone board is used with either brushless DC or microstepping motors, the Option Connector (labeled J5 in Figure 4-1) provides additional signals for multi-phase motor output and input of signals such as Hall sensors. The Option Connector is a single high density 60-pin connector which, if connected to with PMD p/n Cable-7003-01.R, splits (bifurcates) into two 30-pin ribbon cables with headers.

Pin	Connection	Description	Pin	Connection	Description
<b>J5</b>					
1	Vcc	+5V	2	Vcc	+5V
3	GND	Ground	4	GND	Ground
5	PWMMag1A	Phase A PWM magnitude output (axis 1)	6	Hall1 A	Phase A Hall sensor input (axis 1)
7	PWMMag1B	Phase B PWM magnitude output (axis 1)	8	Hall1 B	Phase B Hall sensor input (axis 1)
9	PWMMag1C / PWMSign1B	Phase C PWM magnitude output (axis 1) / Phase B PWM sign output (axis 1)	10	Hall1 C	Phase C Hall sensor input (axis 1)
11	PWMSign1A	Phase A PWM sign output (axis 1)	12	GND	Ground
13	GND	Ground	14	Hall2A	Phase A Hall sensor input (axis 2)
15	PWMMag2A	Phase A PWM magnitude output (axis 2)	16	Hall2B	Phase B Hall sensor input (axis 2)
17	PWMMag2B	Phase B PWM magnitude output (axis 2)	18	Hall2C	Phase C Hall sensor input (axis 2)
19	PWMMag2C / PWMSign2B	Phase C PWM magnitude output (axis 2) / Phase B PWM sign output (axis 2)	20	GND	Ground
21	PWMSign2A	Phase A PWM sign output (axis 2)	22	Hall3A	Phase A Hall sensor input (axis 3)
23	GND	Ground	24	Hall3B	Phase B Hall sensor input (axis 3)
25	PWMMag3A	Phase A PWM magnitude output (axis 3)	26	Hall3C	Phase C Hall sensor input (axis 3)
27	PWMMag3B	Phase B PWM magnitude output (axis 3)	28	GND	Ground
29	PWMMag3C / PWMSign3B	Phase C PWM magnitude output (axis 3) / Phase B PWM sign output (axis 3)	30	Hall4A	Phase A Hall sensor input (axis 4)
31	PWMSign3A	Phase A PWM sign output (axis 3)	32	Hall4B	Phase B Hall sensor input (axis 4)
33	Not used	Not used	34	Hall4C	Phase C Hall sensor input (axis 4)
35	GND	Ground	36	GND	Ground
37	Vcc	+5V	38	Vcc	+5V
39	GND	Ground	40	GND	Ground
41	PWMMag4A	Phase A PWM magnitude output (axis 4)	42	AGND	Ground for analog motor command
43	PWMMag4B	Phase B PWM magnitude output (axis 4)	44	DAC1A	Phase A analog mtr cmd output (axis 1), ±10V
45	PWMMag4C / PWMSign4B	Phase C PWM magnitude output (axis 4) / Phase B PWM sign output (axis 4)	46	DAC2A	Phase A analog mtr cmd output (axis 2), ±10V
47	PWMSign4A	Phase A PWM sign output (axis 4)	48	DAC3A	Phase A analog mtr cmd output (axis 3), ±10V
49	GND	Ground	50	DAC4A	Phase A analog mtr cmd output (axis 4), ±10V
51	AmpEnable1	Amplifier enable signal (Axis 1)	52	DAC1B	Phase B analog mtr cmd output (axis 1), ±10V
53	AmpEnable2	Amplifier enable signal (Axis 2)	54	DAC2B	Phase B analog mtr cmd output (axis 2), ±10V
55	AmpEnable3	Amplifier enable signal (Axis 3)	56	DAC3B	Phase B analog mtr cmd output (axis 3), ±10V
57	AmpEnable4	Amplifier enable signal (Axis 4)	58	DAC4B	Phase B analog mtr cmd output (axis 4), ±10V
59	GND	Ground	60	AGND	Ground for analog motor command

## 4.2.6 Serial Connectors

There are two serial connectors, Serial Connector (J21) and Serial Alt. Connector (J17), each of which provide 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 these two serial connectors, and provide pinouts when operated in RS232 mode, RS485 full duplex mode, and RS485 half duplex mode.



#### 4.2.6.1 Serial Connector (J21)

The connector for J21 is a female DB-9.

Pin	Connection	RS232	RS485 Full Duplex	RS485 Half Duplex
<b>J21</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

#### 4.2.6.2 Serial Alt. Connector (J17)

The J17 connector on the Prodigy/CME Stand-Alone board is a Molex ‘Milli-grid’ style, p/n Molex 87832-1020.

Pin	Connection	RS232	RS485 Full Duplex	RS485 Half Duplex
<b>J17</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	RS485Rcv <sup>+</sup>	no connect	Positive (non-inverting) receive input	no connect
3	Sr1IXmt	Serial 1 transmit output	no connect	no connect
4	Sr12Rcv/RS485Rcv <sup>-</sup>	Serial 2 receive input	Negative (inverting) receive input	no connect
5	Sr1IRcv	Serial 1 receive input	no connect	no connect
6	Sr12Xmt/RS485Xmt <sup>+</sup>	Serial 2 transmit output	Positive (non-inverting) transmit output	Positive (non-inverting) transmit/receive
7	No connect	no connect	no connect	no connect
8	RS485Xmt <sup>-</sup>	no connect	Negative (inverting) transmit output	Negative (inverting) transmit/receive
9	GND	Ground	Ground	Ground
10	GND	Ground	Ground	Ground

### 4.2.7 Sync I/O Connector

The two Sync I/O connectors located on the Prodigy/CME Stand-Alone board (J3 and J4 in [Figure 4-2](#)) allow for the synchronization of multiple Prodigy/CME Stand-Alone boards within a single system. This configuration enables operation within the same cycle period. If multiple boards are installed, yet not inter-connected, any additional boards would begin working after the first board (master) was initialized. However, none of the axes would be synchronized. With Sync I/O activated, the servo loops of all slave boards are synchronized to the servo loop of the master board. This allows for precise synchronization of all implemented axes.

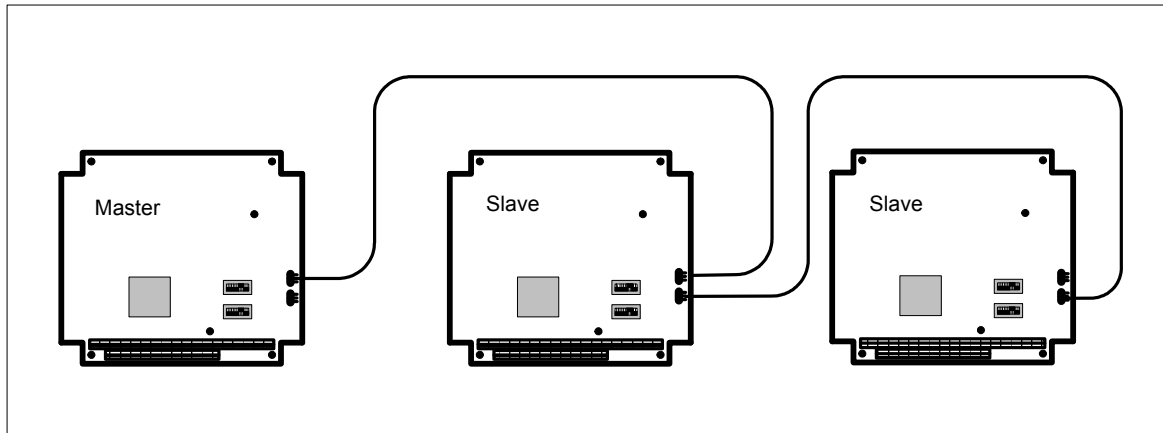
To connect two or more Prodigy/CME Stand-Alone boards for synchronization, a Sync I/O cable (one cable for each set of two boards) is required. This cable may be connected to either of the two Sync I/O connectors on the boards. Both connectors function as either an input or output; the two sync connectors are wired in parallel. For more information on synchronizing multiple Prodigy/CME Stand-Alone boards, see the *Magellan Motion Control IC User Guide*.

The pinouts for the Sync I/O connectors are defined in the following table:

Pin Number	Signal
<b>J3, J4</b>	
1	Sync-in or sync-out pin, depending on whether the board is master or slave.
2	GND

The following diagram shows three synchronized Boards.

**Figure 4-3:**  
Sync I/O  
connector to  
three boards



## 4.2.8 CAN Connectors

The Prodigy/CME Stand-Alone board's controller area network (CAN) transceivers are designed for use with CAN controllers, or with equivalent devices. They are intended 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, J22, and J28, providing electrically identical signals. These two connectors are designed to make it easy to connect the Prodigy/CME Stand-Alone 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 120ohm termination at each end of the bus. Note that it is up to the customer to verify their network topology and operating parameters.

See [Section 2.5.2, "CANbus Communications,"](#) for more information on the functionality of the CANbus port.

The pinouts for both the J22 and J28 CAN connector are as follows:

Pin Number	Signal	Description
<b>J22, J28</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

## 4.2.9 Ethernet Connector

The Prodigy/CME Stand-Alone's Ethernet transceivers are designed for use with 10/100 base-TX Ethernet and support both TCP and UDP protocols. See [Section 2.5.3, "Ethernet Communications,"](#) for more information on the functionality of the Ethernet port.

The Ethernet connector is an 8-pin female RJ45 Category 5 type, 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. No 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

## 4.2.10 Pulse and Direction Connector

This connector (labeled J7 in [Figure 4-1](#)) provides pulse and direction outputs using line drivers. Using this connector with pulse and direction step drivers will provide a higher immunity to noise, reducing the chance of mispositioning. J7 is a 2 by 10 .100 header-style connector. The pins and their associated signals are described in the following table:

Pin Number	Signal	Pin Number	Signal
<b>J7</b>			
1	GND	2	GND
3	axis 4, pulse +	4	axis 4, pulse -
5	axis 1, direction +	6	axis 1, direction -
7	axis 4, direction +	8	axis 4, direction -
9	axis 2, pulse +	10	axis 2, pulse -
11	axis 3, direction +	12	axis 3, direction -
13	axis 1, pulse +	14	axis 1, pulse -
15	axis 2, direction +	16	axis 2, direction -
17	axis 3, pulse +	18	axis 3, pulse -
19	GND	20	GND

## 4.2.11 Power Connectors

J27, J25, and J26 are used to provide power to the board. Only J27 and J26 are intended to be used during production operation of the board, while J25 is primarily intended for table-top development. The J25 connector, which is used with the convenient wall outlet power supply cable set PW-5001-KIT, is installed for the DK (Developer's Kit)-only version of the Prodigy/CME Stand-Alone board. It is not installed for the production versions of the Prodigy/CME Stand-Alone board.

J27 is a 2 pin spring-block style connector. J25 is a two-pin coaxial DC power style connector (ID 2mm, OD 6.3mm), and J26 is a 4-pin Molex KK100 style connector. The table below shows the pin connections for the J27, J25, and J26:

Pin #	J27 Connector	J25 Connector (DK only)	J26 Connector
1	Vcc (+5V)	Vcc (+5V) (Center)	GND
2	GND	GND (Outer)	GND
3	-	-	Vcc (+5V)
4	-	-	Vcc (+5V)



To minimize the risk of unintended machine motion during powerup, care should be taken to control the overall powerup sequence of the Prodigy/CME board and connected system motion amplifiers. The recommended sequence is to first powerup the Prodigy/CME board, followed by powerup of the motion amplifiers. The time delay to ensure stable operation of the control electronics before powering up the amplifiers varies depending on the system's power supplies and configuration, but a typical safe figure is 500 mSec - 1,000 mSec.

## 4.2.12 Connector Parts Reference

The following table is supplied as a reference only.

Label	Description	Board Part Number	Connector Mate
J6	GP Connector	3M P50-100P-S20-EA	3M P50-100S-R1-EA
J5	Option Connector	3M P50-060P-S20-EA	3M P50-060S-R1-EA
J21	Serial Connector	AMP 1-5747150-4	DB-9 Male
J17	Serial Alt. Connector	Molex 87832-1020	Molex 51110-1051 (housing) and 50394-8100 (crimp pins)
J3, J4	Synch Connector	Molex 53398-0271	Molex 51021-0200
J22, J28	CAN Connector	EDAC A00-108-620-450	Male RJ45
J24	Ethernet Connector	EDAC A00-108-620-450	Category 5 Male RJ45
J7	Pulse & Direction Connector	Samtec TSM-110-01-L-DV	Samtec IDSD-10-S
J26	Power Connector	Molex 22-05-3041	Molex 22-01-3047 (housing) and 08-50-0114 (crimp pins)
J27	Power Alt1 Connector	OST ED3000/2	14-28 AWG Wire
J25	Power Alt2 Connector	Kycon KLDX-0202-A	CUI PR-002A

## 4.3 Connections Summary—Motor Amplifiers

The Prodigy/CME Stand-Alone board supports four methods of output to motor amplifiers, as described in the following table:

Motor Output Method	Signal Output
DAC	Analog signals from the onboard D-A converters.
PWM sign-magnitude	Pulse width modulated signals with separate magnitude and sign signals per output phase.
PWM 50/50	Pulse width modulated square-wave signals with a single PWM signal per output phase.
Pulse & Direction	One signal representing pulse information, and a signal representing direction.

In addition, each motor axis may have one, two, or three output phases associated with it. For DC brush motors, the number of phases is one. For multi-phase motors such as brushless DC or microstepping motors, the number of phases can be two or three, depending on the output waveform programmed into the Prodigy/CME Stand-Alone board. For more information, see the *Magellan Motion Control IC User Guide*.

The following tables provide convenient summaries of amplifier connections for common configurations of motor output method and motor type. These outputs should be connected from the designated connector pins to the appropriate amplifier inputs. Note that the names of the pins will vary among amplifiers. Common names are shown.

### 4.3.1 DC Brush Motor Connections

To connect to DC Brush motors the GP Connector (J6) is used.

Motor Output Method	Connection Name	Amplifier Input Connection Name*	Axis 1	Axis 2	Axis 3	Axis 4
DAC	DACI-4A	Ref+ or V+	J6-33	J6-67	J6-34	J6-68
	AGND	Ref- or GND	J6-35	J6-69	J6-36	J6-70
PWM sign/magnitude**	PWMMagI-4A	PWM magnitude	J6-27	J6-61	J5-28	J6-62
	PWMSignI-4A	PWM direction	J6-29	J6-63	J6-30	J6-64
	GND	Ground	J6-15	J6-51	J6-16	J6-52

PMD's Cable-5003-01.R provides a bifurcated 100 pin to dual 50-pin header cable. For your convenience, the following table provides DC Brush motor connections numbered for these cable headers rather than the GP Connector.

“H1” refers to Header 1, and “H2” refers to Header 2 of this cable.

Motor Output Method	Connection Name	Amplifier Input Connection Name*	Axis 1	Axis 2	Axis 3	Axis 4
DAC	DACI-4A	Ref+ or V+	H1-17	H1-34	H2-17	H2-34
	AGND	Ref- or GND	H1-18	H1-35	H2-18	H2-35
PWM sign/magnitude**	PWMMagI-4A	PWM magnitude	H1-14	H1-31	H2-14	H2-31
	PWMSignI-4A	PWM direction	H1-15	H1-32	H2-15	H2-32
	GND	Ground	H1-8	H1-26	H2-8	H2-26

\* Names of amplifier connections vary. Common names are shown.

\*\* Both the Magnitude and Sign connections are used if the Magellan's PWM Sign Magnitude mode is selected. If PWM50/50 mode is selected, then only the Magnitude connections are used.

### 4.3.2 Brushless DC Motor Connections

To connect to Brushless DC motors using internal commutation (provided by the Magellan Motion Control IC), two sinusoidal phase signals are driven per axis via the Option Connector (J5).

Motor Output Method	Connection Name	Amplifier Input Connection Name*	Axis 1	Axis 2	Axis 3	Axis 4
DAC	DACI-4A	Ref1+ or V1+	J5-44	J5-46	J5-48	J5-50
	DACI-4B	Ref2+ or V2+	J5-52	J5-54	J5-56	J5-58
	AGND	Ref- or GND	J5-42	J5-42	J5-60	J5-60
PWM 50/50	PWMMag1-4A	PWM phase 1	J5-5	J5-15	J5-25	J5-41
	PWMMag1-4B	PWM phase 2	J5-7	J5-17	J5-27	J5-43
	PWMMag1-4C	PWM phase 3	J5-9	J5-19	J5-29	J5-45
	GND	Ground	J5-13	J5-23	J5-35	J5-49

PMD's Cable-7003-01.R provides a bifurcated 60 pin to dual 30-pin header cable. For your convenience, the following table provides Brushless DC motor connections numbered for these cable headers rather than the Option Connector. "H1" refers to Header 1, and "H2" refers to Header 2 of this cable.

Motor Output Method	Connection Name	Amplifier Input Connection Name*	Axis 1	Axis 2	Axis 3	Axis 4
DAC	DACI-4A	Ref1+ or V1+	H2-22	H2-23	H2-24	H2-25
	DACI-4B	Ref2+ or V2+	H2-26	H2-27	H2-28	H2-29
	AGND	Ref- or GND	H2-21	H2-21	H2-30	H2-30
PWM 50/50	PWMMag1-4A	PWM phase 1	H1-3	H1-8	H1-13	H1-21
	PWMMag1-4B	PWM phase 2	H1-4	H1-9	H1-14	H1-22
	PWMMag1-4C	PWM phase 3	H1-5	H1-10	H1-15	H1-23
	GND	Ground	H1-7	H1-12	H1-18	H1-25

\* Names of amplifier connections vary. Common names are shown.

### 4.3.3 Microstepping Motor Connections

To connect to step motors in microstepping mode, two sinusoidal phase signals are driven per axis via the Option Connector (J5).

Motor Output Method	Connection Name	Amplifier Input Connection Name*	Axis 1	Axis 2	Axis 3	Axis 4
DAC	DACI-4A	Ref1+ or V1+	J5-44	J5-46	J5-48	J5-50
	DACI-4B	Ref2+ or V2+	J5-52	J5-54	J5-56	J5-58
	AGND	Ref- or GND	J5-42	J5-42	J5-60	J5-60
PWM sign/magnitude**	PWMMag1-4A	PWM magnitude	J5-5	J5-15	J5-25	J5-41
	PWMSign1-4A	PWM direction	J5-11	J5-21	J5-31	J5-47
	PWMMag1-4B	PWM magnitude	J5-7	J5-17	J5-27	J5-43
	PWMSign1-4B	PWM direction	J5-9	J5-19	J5-29	J5-45
	GND	Ground	J5-12	J5-20	J5-28	J5-49

PMD's Cable-7003-01.R provides a bifurcated 60 pin to dual 30-pin header cable. For your convenience, the following table provides microstepping motor connections numbered for these cable headers rather than the Option Connector. "H1" refers to Header 1, and "H2" refers to Header 2 of this cable.

Motor Output Method	Connection Name	Amplifier Input Connection Name*	Axis 1	Axis 2	Axis 3	Axis 4
DAC	DACI-4A	RefI+ or V1+	H2-22	H2-23	H2-24	H2-25
	DACI-4B	Ref2+ or V2+	H2-26	H2-27	H2-28	H2-29
	AGND	Ref- or GND	H2-21	H2-21	H2-30	H2-30
PWM sign/magnitude**	PWMMagI-4A	PWM magnitude	H1-3	H1-8	H1-13	H1-21
	PWMSignI-4A	PWM direction	H1-6	H1-11	H1-16	H1-24
	PWMMagI-4B	PWM magnitude	H1-4	H1-9	H1-14	H1-22
	PWMSignI-4B	PWM direction	H1-5	H1-10	H1-15	H1-23
	GND	Ground	H1-7	H1-12	H1-18	H1-25

\* Names of amplifier connections vary. Common names are shown.

\*\*Both the Magnitude and Sign connections are used if the Magellan's PWM Sign Magnitude mode is selected. If PWM50/50 mode is selected, then only the Magnitude connections are used.

### 4.3.4 Step (Pulse & Direction) Motor Connections

To connect to step motors using single-ended pulse & direction input amplifiers the GP Connector (J6) is used, and for differential drive the Pulse & Direction Connector (J7) as well as the GP Connector (J6) are used.

Motor Output Method	Connection Name	Amplifier Input Connection Name*	Axis 1	Axis 2	Axis 3	Axis 4
Pulse & direction (single-ended)	PulseI-4	Pulse or step	J6-27	J6-61	J6-28	J6-62
	DirectionI-4	Direction	J6-29	J6-63	J6-30	J6-64
	GND	Ground	J6-15	J6-51	J6-16	J6-52
Pulse & direction (differential)	PulseI-4+	Pulse + or step +	J7-13	J7-9	J7-17	J7-3
	PulseI-4-	Pulse - or step -	J7-14	J7-10	J7-18	J7-4
	DirectionI-4+	Direction +	J7-5	J7-15	J7-11	J7-7
	DirectionI-4-	Direction -	J7-6	J7-16	J7-12	J7-8
	GND	Ground	J7-1	J7-2	J7-19	J7-20
Pulse & direction	AtRest I-4	At Rest	J6-33	J6-67	J6-34	J6-68

\* Names of amplifier connections vary. Common names are shown.

## 4.4 Cables

The following table provides a summary of the standard cable accessories available with Prodigy/CME Stand-Alone boards.

Component Part Number	Description
PW-5001-KIT-01.R	Power supply & cable. This is a US Domestic & International wall adapter power supply and associated cable that plugs into the Prodigy/CME Stand-Alone board's and provides power through its power connector (Power Connector J25)

Component Part Number	Description
Cable-5003-01.R	100-pin to dual 50 pin bifurcated ribbon cable. This 3 foot long cable connects to the primary motion connector (GP Connector) and provides two 50 pin header connectors suitable for connection to the IM-1000 breakout module (see below) or other compatible peripherals
Cable-7003-01.R	60-pin to dual 30 pin bifurcated ribbon cable. This 3 foot long cable connects to the Option Connector and provides two 30 pin header connectors.
Cable-4355-01.R	Dual serial cable. This 5 foot long cable connects to the Prodigy/CME Stand-Alone'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 2 meter cable connects to the board's CANbus connector and has RJ45 connectors on both ends.
Cable-RJ45-02-R	Ethernet connector. This 5 foot cable connects to the board's Ethernet connector, has RJ45 connectors on both ends, and is wired in a crossover configuration

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

#### 4.4.1 PW-5001-KIT-01.R

<b>PMD Part #:</b> PW-5001-KIT-01.R	<b>J25 Pin</b>	<b>Signal</b>
<b>Description:</b> 5V, 2A Output AC Wall Adapter power supply, International converter plugs	1 (Center)	Vcc (+5V)
<b>Length:</b> 5.0 feet	2 (Outer)	Ground
<b>Cable:</b> Integrated		



## 4.4.2 Cable-5003-01.R

**PMD Part #:** Cable-5003-01.R

**Description:** Bifurcated high density 100 pin to dual 50-pin header cable.

**Length:** 3.0 feet

**Cable:** (2) 50 wire, 0.050" pitch, 28AWG stranded cables

**Note:** Dual header ends are labeled "Hdr1" and "Hdr2."

J6 Pin	Signal	Connects to	J6 Pin	Signal	Connects to
1	QuadA1+	Hdr1-1	2	QuadA3+	Hdr2-1
3	QuadA1-	Hdr1-2	4	QuadA3-	Hdr2-2
5	QuadB1+	Hdr1-3	6	QuadB3+	Hdr2-3
7	QuadB1-	Hdr1-4	8	QuadB3-	Hdr2-4
9	Index1+	Hdr1-5	10	Index3+	Hdr2-5
11	Index1-	Hdr1-6	12	Index3-	Hdr2-6
13	Vcc	Hdr1-7	14	Vcc	Hdr2-7
15	GND	Hdr1-8	16	GND	Hdr2-8
17	PosLim1	Hdr1-9	18	PosLim3	Hdr2-9
19	NegLim1	Hdr1-10	20	NegLim3	Hdr2-10
21	Home1	Hdr1-11	22	Home3	Hdr2-11
23	GND	Hdr1-12	24	GND	Hdr2-12
25	AxisOut1	Hdr1-13	26	AxisOut3	Hdr2-13
27	PWMMag1A/Pulse1	Hdr1-14	28	PWMMag3A/Pulse3	Hdr2-14
29	PWMSign1A/Direction1	Hdr1-15	30	PWMSign3A/Direction3	Hdr2-15
31	AxisIn1	Hdr1-16	32	AxisIn3	Hdr2-16
33	DAC1A/AtRest1	Hdr1-17	34	DAC3A/AtRest3	Hdr2-17
35	AGND	Hdr1-18	36	AGND	Hdr2-18
37	QuadA2+	Hdr1-19	38	QuadA4+	Hdr2-19
39	QuadA2-	Hdr1-20	40	QuadA4-	Hdr2-20
41	QuadB2+	Hdr1-21	42	QuadB4+	Hdr2-21
43	QuadB2-	Hdr1-22	44	QuadB4-	Hdr2-22
45	Index2+	Hdr1-23	46	Index4+	Hdr2-23
47	Index2-	Hdr1-24	48	Index4-	Hdr2-24
49	Vcc	Hdr1-25	50	Vcc	Hdr2-25
51	GND	Hdr1-26	52	GND	Hdr2-26
53	PosLim2	Hdr1-27	54	PosLim4	Hdr2-27
55	NegLim2	Hdr1-28	56	NegLim4	Hdr2-28
57	Home2	Hdr1-29	58	Home4	Hdr2-29
59	AxisOut2	Hdr1-30	60	AxisOut4	Hdr2-30
61	PWMMag2A/Pulse2	Hdr1-31	62	PWMMag4A/Pulse4	Hdr2-31
63	PWMSign2A/Direction2	Hdr1-32	64	PWMSign4A/Direction4	Hdr2-32
65	AxisIn2	Hdr1-33	66	AxisIn4	Hdr2-33
67	DAC2A/AtRest2	Hdr1-34	68	DAC4A/AtRest4	Hdr2-34
69	AGND	Hdr1-35	70	AGND	Hdr2-35
71	DigitalIn0	Hdr1-36	72	DigitalIn4	Hdr2-36
73	DigitalIn1	Hdr1-37	74	DigitalIn5	Hdr2-37
75	DigitalIn2	Hdr1-38	76	DigitalIn6	Hdr2-38
77	DigitalIn3	Hdr1-39	78	DigitalIn7	Hdr2-39
79	AmpEnable1	Hdr1-40	80	AmpEnable3	Hdr2-40
81	DigitalOut0	Hdr1-41	82	DigitalOut4	Hdr2-41
83	DigitalOut1	Hdr1-42	84	DigitalOut5	Hdr2-42
85	DigitalOut2	Hdr1-43	86	DigitalOut6	Hdr2-43
87	DigitalOut3	Hdr1-44	88	DigitalOut7	Hdr2-44
89	AmpEnable2	Hdr1-45	90	AmpEnable4	Hdr2-45
91	Reset	Hdr1-46	92	AnalogGND	Hdr2-46
93	Analog1	Hdr1-47	94	Analog5	Hdr2-47
95	Analog2	Hdr1-48	96	Analog6	Hdr2-48
97	Analog3	Hdr1-49	98	Analog7	Hdr2-49
99	Analog4	Hdr1-50	100	Analog8	Hdr2-50

### 4.4.3 Cable-7003-01.R

**PMD Part #:** Cable-7003-01.R

**Description:** Bifurcated high density 60 pin to dual 30-pin header cable.

**Length:** 3.0 feet

**Cable:** (2) 30 wire, 0.050" pitch, 28AWG stranded cables

**Note:** Dual header ends are labeled "Hdr1" and "Hdr2."

J5 Pin	Signal	Connects to	J5 Pin	Signal	Connects to
1	Vcc	Hdr1-1	2	Vcc	Hdr2-1
3	GND	Hdr1-2	4	GND	Hdr2-2
5	PWMMag1A	Hdr1-3	6	Hall1A	Hdr2-3
7	PWMMag1B	Hdr1-4	8	Hall1B	Hdr2-4
9	PWMMag1C / PWMSign1B	Hdr1-5	10	Hall1C	Hdr2-5
11	PWMSign1A	Hdr1-6	12	GND	Hdr2-6
13	GND	Hdr1-7	14	Hall2A	Hdr2-7
15	PWMMag2A	Hdr1-8	16	Hall2B	Hdr2-8
17	PWMMag2B	Hdr1-9	18	Hall2C	Hdr2-9
19	PWMMag2C / PWMSign2B	Hdr1-10	20	GND	Hdr2-10
21	PWMSign2A	Hdr1-11	22	Hall3A	Hdr2-11
23	GND	Hdr1-12	24	Hall3B	Hdr2-12
25	PWMMag3A	Hdr1-13	26	Hall3C	Hdr2-13
27	PWMMag3B	Hdr1-14	28	GND	Hdr2-14
29	PWMMag3C / PWMSign3B	Hdr1-15	30	Hall4A	Hdr2-15
31	PWMSign3A	Hdr1-16	32	Hall4B	Hdr2-16
33	Not used	Hdr1-17	34	Hall4C	Hdr2-17
35	GND	Hdr1-18	36	GND	Hdr2-18
37	Vcc	Hdr1-19	38	Vcc	Hdr2-19
39	GND	Hdr1-20	40	GND	Hdr2-20
41	PWMMag4A	Hdr1-21	42	AGND	Hdr2-21
43	PWMMag4B	Hdr1-22	44	DAC1A	Hdr2-22
45	PWMMag4C / PWMSign4B	Hdr1-23	46	DAC2A	Hdr2-23
47	PWMSign4A	Hdr1-24	48	DAC3A	Hdr2-24
49	GND	Hdr1-25	50	DAC4A	Hdr2-25
51	AmpEnable1	Hdr1-26	52	DAC1B	Hdr2-26
53	AmpEnable2	Hdr1-27	54	DAC2B	Hdr2-27
55	AmpEnable3	Hdr1-28	56	DAC3B	Hdr2-28
57	AmpEnable4	Hdr1-29	58	DAC4B	Hdr2-29
59	GND	Hdr1-30	60	AGND	Hdr2-30

### 4.4.4 Cable-4355-01.R

**PMD Part #:** Cable-4355-01.R

**Description:** Male DB-9 Dual Serial to two single-channel female DB-9 cable

**Length:** 5.0 feet

**Cable:** (2) 3 conductor, 26AWG, Untwisted, shielded, UL STYLE 2464

**Note:** Dual female DB-9 ends are labeled "Port1" and "Port2."

J21 Pin	Signal	Connects to
1	RS485Select	-
2	Sr1IXmt	Port1-2
3	Sr1IRcv	Port1-3
4	No connect	-
5	GND	Port1-5, Port2-5
7	Sr12Rcv/ RS485Rcv <sup>-</sup>	Port2-3
8	Sr12Xmt/ RS485Xmt <sup>+</sup>	Port2-2
9	RS485Xmt <sup>-</sup>	-

### 4.4.5 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		
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	8

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

### 4.4.6 TRM-RJ45-02-R

**PMD Part #:** TRM-RJ45-02-R

**Description:** RJ45 CANbus terminator  
(120 ohm between pins 1 and 2)

## 4.5 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)
Power requirements:	4.8V to 5.25V operating range, 0.8A (no outputs on)
Supply voltage limits:	-0.3V to +7V
Analog (DAC) output range:	-10.0V to +10.0V, $\pm 3\text{mA}$ min/axis, short circuit protected
Analog input range:	0 to 3.3V, 1.4 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)

## 4.6 Certifications & Compliance

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



This page intentionally left blank.

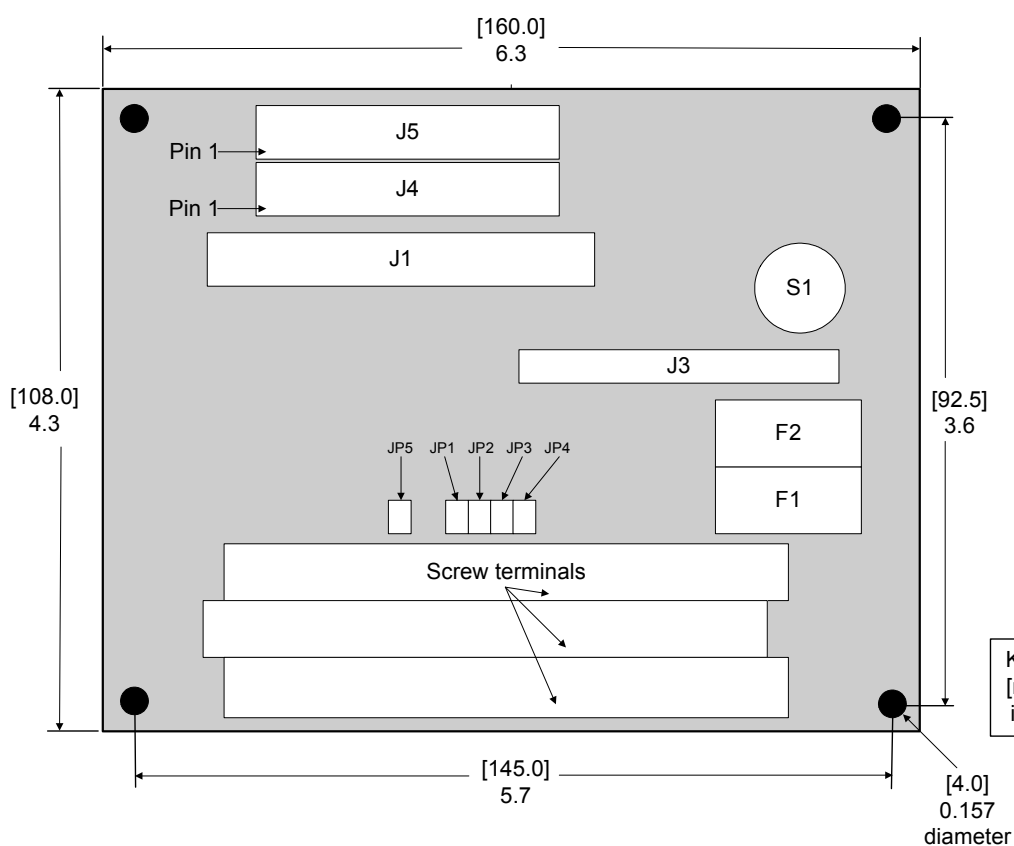
# 5. Interconnect Module

## In This Chapter

- ▶ IM-1000 Interconnect Module
- ▶ IM-600 Interconnect Module

## 5.1 IM-1000 Interconnect Module

The IM-1000 is an interconnect module which assists in the set up and configuration of Prodigy/CME Stand-Alone boards, and also provides complete options for external connections. All wiring to and from the Prodigy/CME Stand-Alone board, the amplifiers, power supplies, IOs and encoder feedback may easily be connected to this central point.



**Figure 5-1:**  
IM-1000  
location of  
components

Connector	Description
F1	1 Amp fuse for power supply
F2	1 Amp fuse for power supply
J1	100-position connector

Connector	Description
J3	OPTO22 connector
J4	50-position connector
J5	50-position connector
S1	External hardware reset button
Screw terminals	(3) 100-terminal connection blocks for connecting to one 100-position, or two 50-position cables.

The enclosure of the board is a standard Phoenix DIN rail mounting system. Alternatively, the enclosure may be removed, and the board mounted to other systems via the 3.5 mm mounting holes located on the corners of the board.

The pinout descriptions that follow provide a detailed illustration of the IM-1000's connections. Please observe the differences between the DC brush and brushless DC and stepping motor versions. The screw terminal numbers correspond to the pinout description for Prodigy/CME Stand-Alone GP Connector (J6) Cable-5003 Header connectors (see [Section 4.4.2, "Cable-5003-01.R,"](#) for a description). "H1" in the tables below refers to Header1, and "H2" in the tables refers to Header2.

Header	IM-1000	Screw Terminal
H1	J4	1-50
H2	J5	51-100

The following table lists the functions of the remaining screw terminals:

Screw Terminals	Function
101, 103	5V from J1 through 1 Amp fuse F2
102, 104, 105	GND
106	External power in
107	GND
108	Unused

The Cable-5003 Header 1 and Header 2 cables supply 5V DC to pins H1-7, H1-25, H2-7, and H2-25 through a 1 Amp fuse to power encoders. When the jumpers JP1, JP2, JP3 and JP4 are installed 1-2, this 5V is passed on to screw terminals 7, 25, 57, and 75, respectively. Alternatively, it is also possible to use an external power supply in the range of 5 - 12 VDC for the encoders. To do this, install jumpers JP1, JP2, JP3 and JP4 on pins 2-3 and connect the external power to screw terminal 106 and its ground to 107. This external power will be limited to 1 Amp by fuse F1.

Connections to J3 (the OPTO22 connector) are as follows:

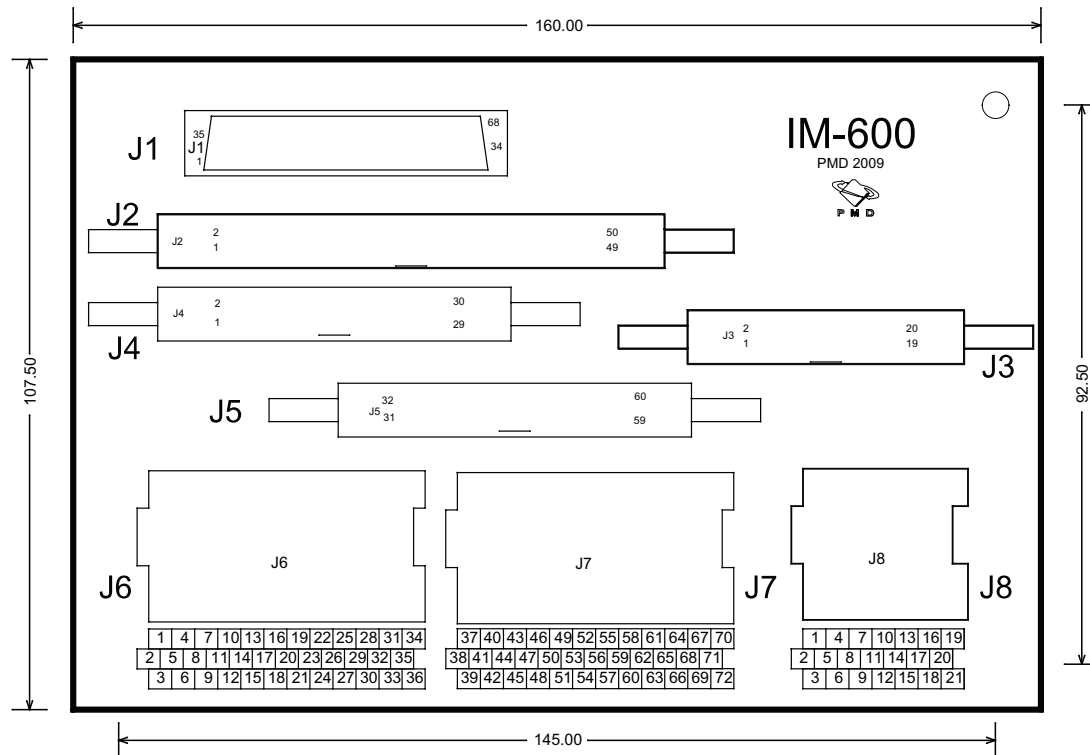
H1/H2	J3	Description
	49	5V if JP5 jumper is installed
H1-36	47	DigitalIn0
H1-37	45	DigitalIn1
H1-38	43	DigitalIn2
H1-39	41	DigitalIn3
H2-36	39	DigitalIn4
H2-37	37	DigitalIn5
H2-38	35	DigitalIn6
H2-39	33	DigitalIn7
H1-41	31	DigitalOut0
H1-42	29	DigitalOut1
H1-43	27	DigitalOut2
H1-44	25	DigitalOut3
H2-41	23	DigitalOut4
H2-42	21	DigitalOut5
H2-43	19	DigitalOut6
H2-44	17	DigitalOut7
H1-40	15	AmpEnable1
H1-45	13	AmpEnable2
H2-40	11	AmpEnable3



H1/H2	J3	Description
H2-45	09	AmpEnable4
H1-13	07	AxisOut1
H1-30	05	AxisOut2
H2-13	03	AxisOut3
H2-30	01	AxisOut4
All even-numbered pins		GND

## 5.2 IM-600 Interconnect Module

The IM-600 is an interconnect module which assists in the set up and configuration of the Option Connector and the Pulse & Direction Connector for all versions of Prodigy boards by providing jack screw 'breakout' functionality, making it easy to hand-connect motion peripherals.



**Figure 5-2:**  
IM-600  
location of  
components

The following table summarizes the functionality of the various components of the IM-600 module.

Connector	Description
J1	68-pin SCSI-style connector
J2	50-pin dual header connector
J3	20-pin dual header connector
J4 & J5	30-pin dual header connectors
J6/J7	72-pin jack screw array
J8	21-pin jack screw array

The enclosure of the board is a standard Phoenix DIN rail mounting system. Alternatively, the enclosure may be removed, and the board mounted to other systems via the 3.5 mm mounting holes located on the corners of the board.

The table below illustrates the connections from the PC/104, PCI, and Stand-Alone versions of the Prodigy board's Option Connector and Pulse & Direction Connector, to the IM-600's connectors. In all cases the jack screw pins are wired 'straight through' from the connector input, so that the documentation for the Prodigy board connectors or

cable headers can be conveniently used to determine correct pin placement on the IM-600's jack screw pin connections:

Prodigy Board format	Board Connector & Type	Connect to IM-600 Connector	Screw terminal connections & wiring
PC/I04*	J9 - Option Connector	J2	J6/J7 pins 1-50, straight through
	J12 - Pulse & Direction Connector	J3	J8 pins 1-20, straight through
PCI*	J5 - Option Connector	J1	J6/J7 pins 1-68, straight through
Stand-Alone	J5/H1** - Option Connector	J4	J6/J7 pins 1-30, straight through
	J5/H2** - Option Connector	J5	J6/J7 pins 31-60, straight through
	J7 - Pulse & Direction Connector	J3	J8 pins 1-20, straight through

\* Both CME and non-CME version Prodigy boards

\*\* J5 Option Connector on the Prodigy/CME Stand-Alone board bifurcates via PMD p/n Cable-7003-01.R into two 30-pin dual headers, labeled Header 1 (H1) and Header 2 (H2)

# Index

## Numerics

100-terminal connection blocks 88

50-position connector 88

## A

accessing

communication ports 55

Magellan-attached devices 59

on-card resources 58

accessory products 10

actions, PRP 55

AmpEnable 19

amplifier

connections 77

enable output signals 35

inputs 77

amplifiers 68

analog

input 50

input range 84

output range 84

applying power 19

automatically assigned addresses 58

AxisIn sensor 49

AxisOut sensor 49

## B

brushless DC motor connections 16, 78

## C

Cable-4355-01.R 83

Cable-5003-01.R 81

Cable-7003-01.R 82

Cable-RJ45-02-R 83

cables 79

CAN connector 74, 89

CAN serial communication physical layer 74

card

function summary 30

types 9

communication ports, accessing 55

components table, front of board 14

connection summary 16, 76

connector 14

functions 68

parts reference 76

pins 77

connectors 68

controller area network transceivers 74

cycle period 73

## D

DAC 76

out 51

output enable 36

DC brush motor connections 16, 77

differential

connections 66

encoding 49

receive capability 74

transmit capability 74

digital I/O, general purpose 34

digital outputs drive capacity 84

## E

encoder 68

connections 66

feedback 87

inputs 66

settings 66

signals 66

environmental and electrical ratings 84

Ethernet connector 75

external

connections 87

hardware reset button 88

## F

first time system verification 20

front diagram 14

## G

general-purpose

digital I/O 34

inputs 34

outputs 34

GP connector 68

## **H**

Hall sensors 72  
Home sensor 49

## **I**

IM-1000 87  
index capture 32  
Index signal 49  
inputs, general-purpose 34  
installation 14  
    sequence 11  
interconnect module 87  
ISO 11898 standard 74

## **L**

Limits sensor 49

## **M**

Magellan instruction set 32  
message URL <https://www.pmdcorp.com/company/patents> iii  
microstepping  
    cards 70  
    motor connections 17, 78  
motion processor reset 37  
motor  
    amplifier output methods 76  
    axis 77  
    output configuration 15  
    output method 77  
multi-phase  
    motor output 72  
    motors 77

## **N**

negative direction limit input 49  
noise immunity 75

## **O**

operating temperature 84  
option connector 72  
OPTO22 connector 88  
output  
    phases 77  
    waveform 77  
outputs, general-purpose 34

## **P**

peripheral connections 56  
peripherals 68  
Phoenix DIN rail mounting system 88, 89  
Phoenix EN rail mounting system 88, 89  
PMD Resource Access Protocol 53  
polarity 49  
positive direction limit input 49  
power  
    connector 75  
    requirements 84  
power up 19  
Prodigy/CME Stand-Alone connectors 14  
profile generation 30  
PRP 53  
    actions 55  
    communication formats 61  
    resources 54  
pulse & direction 50  
    cards 71  
    connector 75  
    differential output connector 75  
PW-5001-KIT 80  
PWM  
    50/50 76  
    out 50  
    sign-magnitude 76

## **Q**

QuadA signal 49  
QuadB signal 49

## **R**

ReadIO command 34  
required hardware 11  
reset  
    condition 19, 37  
    monitor word 37  
resistor pack 14  
    settings 15, 66  
resistor packs 65, 66  
resource  
    addressing 53  
    PRP 54  
Resource Access Protocol 53

## **S**

screw terminal 88  
serial

---

- communications 72
- connector 72
- servo loop closure 30
- shunting 36
- signal conditioning 31
- single-ended
  - connections 49
  - encoders 66
- software
  - libraries 51
- step (pulse & direction) motor connections 79
- storage temperature 84
- supply voltage limits 84
- sync I/O 73
  - cable 73
  - connectors 73
  - pinouts 74
- synchronizing multiple boards 73

## **T**


TRM-RJ45-02-R 83

## **U**

- under-voltage detection circuit 37
- undesired motion 19
- user I/O memory map 85
- user-settable components 65

## **W**

- watchdog function 36
- WriteIO command 34



---

This page intentionally left blank.

---

For additional information, or for technical assistance,  
please contact PMD at (978) 266-1210.

You may also e-mail your request to [support@pmdcorp.com](mailto:support@pmdcorp.com)

Visit our website at <http://www.pmdcorp.com>



Performance Motion Devices  
1 Technology Park Drive  
Westford, MA 01886