Magellan® Motion Control IC

DK58113 Developer Kit
User Manual

Revision 1.5 / March 2025

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 PMD.

Copyright 1998–2025 by Performance Motion Devices, Inc.

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

Warranty

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

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

Unauthorized Applications

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

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

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

Disclaimer

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

Patents

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

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

Related Documents

Magellan® Motion Control IC User Guide

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

MC58113 Electrical Specifications

Information on physical and electrical characteristics, timing diagrams, pin descriptions, application notes and application schematics of MC58113 IC.

Atlas Digital Amplifier User Manual

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

Atlas Digital Amplifier Complete Technical Reference

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

C-Motion® Engine Development Tools Manual

Complete guide to developing software for PMD controller products. Includes examples on how to use the C-Motion Magellan, C-Motion PRP, and C-Motion PRP II C-language libraries, and details the recommended code development sequence for microcontroller-based, PC-based , and C-Motion Engine-based systems.

C-Motion Magellan Programming Reference

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

Table of Contents

List of Figures

1. Installation

In This Chapter

▶ Introduction

▶ Magellan Motion Control IC Family Overview

▶ DK58113 Board

▶ Developer Kit Cables & Accessories

▶ Guide to this Manual

▶ Software Installation

▶ Recommended Hardware

1.1 Introduction

The PMD DK58113 Developer Kit is an integrated board/software package that serves as an electrical and software design tool for building systems that use Magellan MC58113-series ICs.

Five DK58113 developer kit variations support all members of the MC58113 IC family, as shown below:

Developer Kit p/n	Installed IC*	Motors supported	Comments
DK58113	MC58113	DC Brush, Brushless DC, step motor	
DK58113S	MC58113S	DC Brush, Brushless DC, step motor	Socketed version of DK58113
DK53113	MC53113	Brushless DC	
DK51113	MC51113	DC Brush	
DK54113	MC54113	Step motor	

* See Section 1.3.1, "Installed MC58113 IC Version" for additional information on installed IC version.

All of the above Developer Kit versions share the same physical DK58113 board. They differ in the specific type of MC58113-series IC chip that is installed in the board and in whether the MC58113 IC is socketed or not.

Note that throughout this manual the term MC58113 may be used to mean all members of the MC58113 series including the MC58113, MC53113, MC51113, and MC54113 ICs. The term DK58113 may be used to mean to all members of the DKs including the DK58113, DK53113, DK51113, and DK54113 developer kits.

1.2   Magellan Motion Control IC Family Overview

The following table presents a feature summary of the products in the Magellan Motion Control IC product family:

	MC58000 Series (Except MC58113)	MC55000 Series	MC58113 Series
# of axes	1, 2, 3, 4	1, 2, 3, 4	1+ (primary & aux channel encoder input)
Motor types supported	DC Brush, Brushless DC, step motor	Step motor	DC Brush, Brushless DC, step motor
Output format	SPI Atlas, PWM, DAC, Pulse & direction	Pulse & direction	SPI Atlas, PWM, DAC, Pulse & direction
Parallel host communication	✓	✓	
Serial host communication	✓	✓	✓
CAN 2.0B host communication	✓	✓	✓
SPI host communication			✓
Incremental encoder input	✓	✓	✓
Parallel word device input	✓	✓	
Index & Home signals	✓	✓	✓
Position capture	✓	✓	✓
Directional limit switches	✓	✓	✓
PWM output	✓		✓
Parallel DAC output	✓		
SPI Atlas interface	✓		✓
SPI DAC output	✓		✓
Pulse & direction output	✓	✓	✓
Digital current control	✓ (with Atlas)		✓
Field oriented control	✓ (with Atlas)		✓
Under/overvoltage sense	✓ (with Atlas)		✓
I^2T Current foldback	✓ (with Atlas)		✓
DC Bus shunt resistor control			✓
Overtemperature sense	✓ (with Atlas)		✓
Short circuit sense	✓ (with Atlas)		✓
Ground fault detection	✓ (with Atlas)		
Trapezoidal profiling	✓	✓	✓
Velocity profiling	✓	✓	✓
S-curve profiling	✓	✓	✓
Electronic gearing	✓	✓	✓
On-the-fly changes	✓	✓	✓
PID position servo loop	✓		✓
Dual biquad filters	✓		✓
Dual encoder loop	✓ (multi-axis configurations only)		✓
Programmable derivative sampling time	✓		✓
Feedforward (accel & vel)	✓		✓
Data trace/diagnostics	✓		✓
Motion error detection	✓	✓ (with encoder)	✓
Axis settled indicator	✓	✓ (with encoder)	✓
Analog input	✓	✓	✓
Programmable bit output	✓	✓	✓
Software-invertible signals	✓	✓	✓
User-defined I/O	✓	✓	
Internal Trace Buffer			✓
External RAM support	✓	✓	
Multi-chip synchronization	✓		✓

	MC58000 Series (Except MC58113)	**MC55000 Series**	**MC58113 Series**
Chipset configurations	MC58420 (4 axes, 2 ICs)	MC55420 (4 axes, 2 ICs)	MC51113 (1+ axis, 1 IC)
	MC58320 (3 axes, 2 ICs)	MC55320 (3 axes, 2 ICs)	MC53113 (1+ axis, 1 IC)
	MC58220 (2 axes, 2 ICs)	MC55220 (2 axes, 2 ICs)	MC54113 (1+ axis, 1 IC)
	MC58120 (1 axis, 2 ICs)	MC55120 (1 axis, 2 ICs)	MC58113 (1+ axis, 1 IC)
	MC58110 (1 axis, 1 IC)	MC55110 (1 axis, 1 IC)	
IC Package: CP chip	MC58x20: 144 pin TQFP	MC55x20: 144 pin TQFP	100 pin TQFP
	MC58110: 144 pin TQFP	MC55110: 144 pin TQFP	
IC Package: IO chip	MC58x20: 100 pin TQFP	MC55x20: 100 pin TQFP	N/A
	MC58110: NA	MC55110: NA	
Motion control IC developer kit p/n's	DK58420	DK55420	DK51113
	DK58320	DK55320	DK53113
	DK58220	DK55220	DK54113
	DK58120	DK55120	DK58113
	DK58110	DK55110	DK58113S

1.3   DK58113 Board

The heart of the DK58113 Developer Kit is the DK58113 printed circuit board that contains interface and amplifier circuitry to allow various features of the MC58113-family ICs to be exercised. Here is a summary of the features provided by the DK58113 board:

- Supports DC Brush, Brushless DC, and step motors

- Socketed version of DK (DK58113S) allows MC58113s to be swapped out for testing or user configuration storage

- High performance on-board amplifier with current feedback supports all motor types

- Interfaces to external PMD Atlas amplifier, user-designed switching amplifier, or pulse & direction amplifier

- RS232, RS485, CANbus, and SPI (Serial Peripheral Interface) host communications

- Single DC-voltage supply

- Primary and auxiliary axis quadrature signal input with Index and Home capture

- Hall sensor, Home, limits, AxisIn and AxisOut signals

- Support for overtemperature, overcurrent, over and undervoltage sense

- High current external shunt resistor support

- Pulse & Direction output signals with AtRest for use with external step motor amplifiers

- Compact 3.3" x 4.7" standalone form factor (8.4 cm x 11.9 cm)

1.3.1   Installed MC58113 IC Version

MC58113 family ICs, in addition to the base part number which is listed in the table in Section 1.1, "Introduction" are also identified with a version number. Version numbers represent revisions of the base part number, generally for the purpose of implementing functional corrections. This version information appears as digits or characters after the base part number.

To check the version of the installed MC58113 family IC in your developer kit board view the Project window from Pro-Motion. The full part number of the MC58113 IC will be listed after "Model" when Pro-Motion is connected to

the DK58113 board. When you later order motion ICs for development of a user-designed board or when the board goes into production it is recommended to order the same version.

You can order specific motion IC part numbers including version directly from PMD. If you order PMD motion ICs from outlets such as Digikey or Mouser generally only one version is available and therefore you should be aware that the version number may be different. The version should be indicated in the Description of the motion IC on those websites.

1.4   Developer Kit Cables & Accessories

In addition to the DK58113 board all DK58113 developer kits come with the following cables and accessories.

PMD Component Part Number	Description
Cable-USB-DB9	USB to DB9 serial cable.
Cable-RJ45-02	RJ45 to RJ45 connector. This cable connects to the board's CANbus connectors.
TRM-RJ45-02	120 ohm CANbus terminator.
MC-HW-05	DB15 terminal screw expander board.
CONN-0122-11	2-pin power & shunt connector terminal screw plug (qty 2).
CONN-0121-11	5-pin motor drive connector terminal screw plug.

1.5   Guide to this Manual

This manual is designed to help you get your motion hardware setup connected and operating with the DK58113 Developer Kit board. In addition, this manual shows you how to continue with development of your application by further exercising the connected motor hardware, optimizing the motion system control parameters, and developing software for the production control application.

Here is a summary of the content in the remaining chapters of this manual:

Chapter 2, *Quick Start Guide*, provides instructions on connecting and verifying proper functioning of your motion hardware with the Dk58113 board.

Chapter 3, *Going Further with Pro-Motion*, describes the most frequently used features of Pro-Motion. You will find the content in this chapter helpful if you are new to Pro-Motion and PMD's Magellan Motion Control ICs.

Chapter 4, *DK58113 Board Operation*, provides a complete reference for operation of the board including how to access board functions via Pro-Motion.

Chapter 5, *Communication Port Connections*, provides an overview of how to develop software application code for your PMD-based control system.

Chapter 6, *Software Development*, provides reference information that you may find useful in connecting the machine controller board to your motion hardware.

Appendix A, *DK58113 Board Schematic*, provides complete schematics for the DK58113 board.

1.6 Software Installation

The software distribution for the DK58113 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. There is a *ReadMe.txt* file that may contain additional useful information. When ready execute the **Pro-Motion** install. Pro-Motion is a Windows application that will be used to communicate with and exercise your developer kit.

5 You can also extract one or both of the following SDKs (Software Development Kit) used with the DK58113 board.

- **C-Motion Magellan SDK** – an SDK for creating C-language user applications for PMD products that utilize a Magellan or Juno formatted protocol

- **C-Motion PRP SDK** – An SDK for creating PC-based applications using .NET (C#, VB) programming languages, and for creating C-language user applications for PMD products that utilize a Magellan/Juno or PRP (PMD Resource Access Protocol) formatted protocol

The next few sections give more information on each of these items.

1.6.1 Pro-Motion

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

- Motion oscilloscope graphically displays motion control parameters in real-time

- AxisWizard to automate axis setup and configuration

- 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 console window

- C-Motion Engine user application code download

For more information on Pro-Motion see Chapter 3, *Going Further with Pro-Motion*.

1.6.2   C-Motion

C-Motion provides a convenient set of callable routines comprising the C language code required for controlling Magellan ICs. C-Motion includes the following features:

- Magellan axis virtualization

- Ability to communicate to multiple PMD motion boards 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 board, or C-Motion Engine

- Can be easily linked to any C/C++ application

There are three different versions of C-Motion; C-Motion Magellan, C-Motion PRP, and C-Motion PRP II. C-Motion Magellan is used with PMD products that utilize a direct Magellan or Juno formatted protocol. C-Motion PRP is used in systems that support both Magellan/Juno protocol devices and PRP (PMD Resource Access Protocol) protocol devices. C-Motion PRP is also used in motion applications that will use the .NET (C#, VB) programming languages. C-Motion PRP II is used with ION/CME N-Series Digital Drives.

For more information on C-Motion see Chapter 6, *Software Development*.

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 boards or modules

- Ability to communicate via PC/104 bus, serial, CAN, Ethernet, SPI, or 8/16-bit parallel bus

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

1.7   Recommended Hardware

To install a DK58113 board the following hardware is recommended. Note that this list assumes that the on-board amplifier will be used, which is the configuration used in the Quick Start Guide.

- 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.

- DC Brush, Brushless DC, or step motor according to the developer kit purchased. For example DK58113 developer kits can support all three motors types, DK51113 developer kits support DC Brush motors, etc. Encoder feedback is a requirement for DC Brush motors, and is normally used with Brushless DC motors (although not required because Hall sensors can be used for position feedback). For step motors, encoders are optional.

- Cables as required to connect to the motor and associated motion hardware such as feedback signals, home sensor, and limit switches. If the auxiliary axis is being used, then additional cables will be used to connect to this second encoder.

- DC power supply which can have a voltage from 12 to 56 volts. The DK58113 board requires only a single voltage supply. The board logic and other circuitry is powered from this input voltage using an on-board DC to DC converter.

This page intentionally left blank.

2. Quick Start Guide

In This Chapter

▶ Step #1 — Configuring the Board

▶ Step #2 — Making Motion Hardware Connections

▶ Step #3 — Applying Power

▶ Step #4 — First Time System Verification

▶ Next Steps

Here are the steps you will execute to set up the DK58113 board so that it can control your motion hardware successfully. If you haven't already installed the software you should do this now. See Section 1.6, "Software Installation," for instructions on installing the software.

Step 1 The first step is to configure the board. See Section 2.1, "Step #1 — Configuring the Board," for a description.

Step 2 Next connect your system's encoder(s), motion peripherals, motors, power supply, and PC to the DK58113 board. See Section 2.2, "Step #2 — Making Motion Hardware Connections," for details.

Step 3 Next you will provide the board with power. See Section 2.3, "Step #3 — Applying Power," for more information.

Step 4 The final step to is to perform a functional test of the finished system. See Section 2.4, "Step #4 — First Time System Verification," for a description of this procedure.

Once these steps have been accomplished setup is complete and the board and connected motion hardware are ready for operation.

2.1 Step #1 — Configuring the Board

Figure 2-1 shows the location of the principle on-board components of the DK58113 board.

**Figure 2-1:
DK58113
Board
Components
Location**
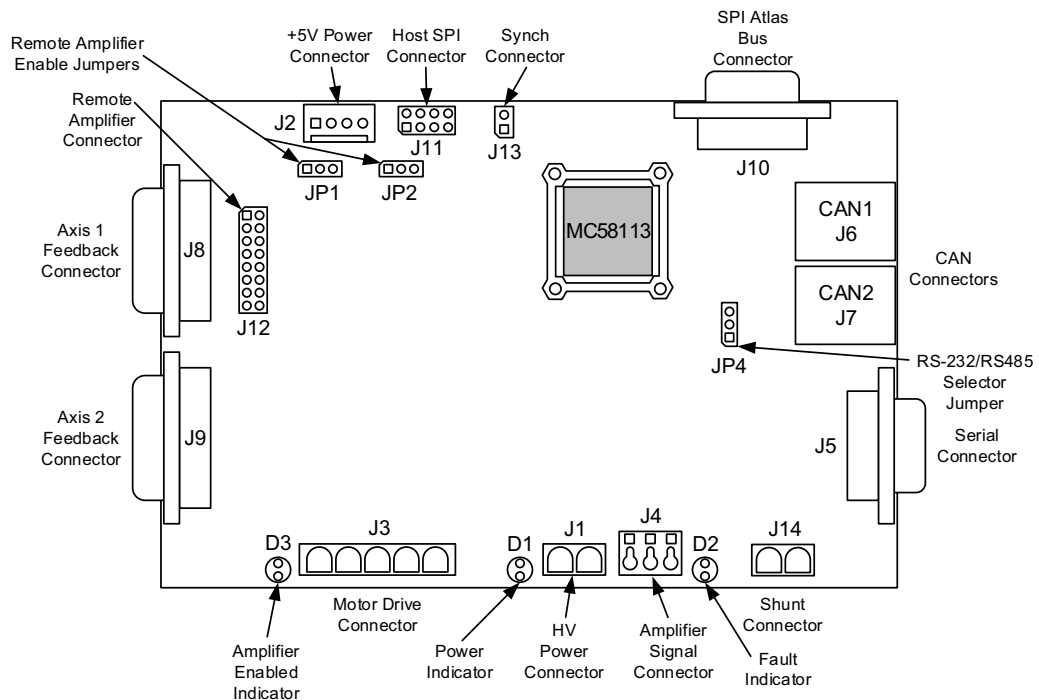
The following table describes these components:

Label	Description
J1	HV Power Connector
J3	Motor Drive Connector
J8	Axis 1 Feedback Connector
J9	Axis 2 Feedback Connector (auxiliary axis)
J4	Amplifier Signal Connector
J10	SPI Atlas Bus Connector
J12	Remote Amplifier Connector
J14	Shunt Connector
J6, J7	CAN1, CAN2 Connectors (respectively)
J5	Serial Connector
J11	Host SPI Connector
J13	Synch Connector
J2	+5V Power Connector
D1, D2, D3	Power, fault, and amplifier enabled LED indicators (respectively)
JP1, JP2	Remote amplifier enable jumpers
JP4	RS232/RS485 selector jumper

2.1.1 Jumper Settings

There are no jumper changes that need to be made for the board to be compatible with the instructions contained in this chapter. However, for reference the table below shows the available jumper settings of the DK58113 board:

Jumper ID	Factory Default Setting	Setting &Description	
JP1, JP2	1-2 (on-board amplifier)	1-2	Installing jumpers at 1-2 for JP1 and JP2 configures the DK58113 for operation of the on-board amplifier.
		2-3	Installing jumpers at 2-3 for JP1 and JP2 disables the on-board amplifier, and configures the DK58113 for operation with a user-designed amplifier via the J12 Remote Amplifier Connector, or with an Atlas DK amplifier via the J10 connector.
JP4	1-2 (RS232)	1-2	Installing a jumper at 1-2 for JP4 configures the DK58113 for RS232 serial operation.
		2-3	Installing a jumper at 2-3 for JP4 configures the DK58113 for RS485 serial operation.

2.1.2 Enable Signal

The MC58113 requires an active Enable signal to operate. To accomplish this the Amplifier Signal Connector (J4) is used. Connect terminal #1 of J4 to terminal #3 of J4 using a short wire. J4 provides convenient push-type connections, so no other hardware is needed to make this connection.

For reference the following table provides the pinouts of the J4 terminal block connector:

Setting & Description	Pin #	Description
J4 — Amplifier Signal Connector		
Enable	1	Enable input. Must be tied low (GND) to enable the MC58113 for full operation.
FaultOut	2	Programmable FaultOut signal output.
GND	3	Digital ground

2.2 Step #2 — Making Motion Hardware Connections

The next few sections detail how to make the needed connections between the DK58113 board and your motion hardware, PC, and power supply.

2.2.1 Encoder & Motion Peripheral Connections

The following table summarizes encoder and motion peripheral signal connections to the DK58113 board. All connections are made through the Axis Feedback Connector for axis #1 (J8), which is a high density female DB-15. Although you can create your own DB-15 cable to connect these signal wires, many users will use the DB-15 terminal screw breakout board included with the developer kit for convenient terminal screw connection of signal wires.

Encoders are required for controlling the position of DC Brush and Brushless DC motors. Encoders are optional for step motors. Quadrature encoders can use a differential wiring scheme where each signal (quadA, quadB, and index) uses a positive (+) and negative (-) connection, or they can use a single-ended scheme with a single connection per signal. If available use of differential connections is highly recommended. If single-ended encoders are used they are

connected to the + differential signal. In addition to quadA, quadB, and index signals encoders also typically require 5V and GND connections.

Hall signals are used with Brushless DC motors only. Although they are not required, they should be used if available. Home and position limit sensors are optional.

Pin #	Signal Name	Description
		J8 — Axis 1 Feedback Connector
1	QuadA1+	Differential A+ quadrature input. *optional for step motor axes*
2	QuadA1-	Differential A- quadrature input. *optional for step motor axes*
3	QuadB1+	Differential B+ quadrature input. *optional for step motor axes*
4	QuadB1-	Differential B- quadrature input. *optional for step motor axes*
5	GND	This is the preferred ground connection for the quadrature and Index signal inputs
6	Index1+	Differential Index+ quadrature input. *optional for step motor axes*
7	Index1-	Differential Index- quadrature input. *optional for step motor axes*
8	Hall1A	Hall signal input phase A. *not used for DC Brush or step motors*
9	Hall1B	Hall signal input phase B. *not used for DC Brush or step motors*
10	Hall1C	Hall signal input phase C. *not used for DC Brush or step motors*
11	Home1	Home signal input (*optional*)
12	PosLim1	Positive position limit input (*optional*)
13	NegLim1	Negative position limit input (*optional*)
14	+5V	+5V power output which may be used to power the motor's encoder circuitry
15	NC	No Connect

For additional electrical information on any of these signals refer to Section 4.8, "Motion Peripheral Signals."

2.2.1.1   Auxiliary Encoder Input

For this 'quick start' guide the auxiliary encoder inputs which are located on connector J9, will not be used. For detailed information on the J9 connector see Section 7.2.3, "Axis Feedback Connectors (J8, J9)."

2.2.2   Motor Coil Connections

The following table summarizes the motor drive connections from the DK58113 to the coils (also called windings) of the motor. The Motor Drive Connector, J3, provides these connections and is designed to connect to all supported motor types: Brushless DC, DC Brush, and step motor. There are four motor drive connections and a shield connection. Not every motor type uses all four drive connections however.

The J3 Motor Drive Connector is a male Molex Mini-Fit Plus style connector.

Pin #	Signal Name	Description
		J3 — Motor Drive Connector
1	MotorA	A motor drive lead. Used with all motor types.
2	MotorB	B motor drive lead. Used with all motor types
3	MotorC	C motor drive lead. Used with all motor types except DC Brush
4	MotorD	D motor drive lead. Use with step motors only
5	Case/shield	Connection to motor case/shield. A shield connection is strongly recommended for most motor setups

The table shows which leads should be connected for each supported motor type:

Motor type	Motor Drive Connector Signal	Motor Coil Connections
Brushless DC	Pin #1 — MotorA	A winding connection
	Pin #2 — MotorB	B winding connection
	Pin #3 — MotorC	C winding connection
	Pin #5 — Case/shield	(optional) motor shield connection
DC Brush	Pin #1 — MotorA	+ winding connection
	Pin #2 — MotorB	- winding connection
	Pin #5 — Case/shield	(optional) motor shield connection
Step motor	Pin #1 — MotorA	phase A+ winding connection
	Pin #2 — MotorB	phase A- winding connection
	Pin #3 — MotorC	phase B+ winding connection
	Pin #4 — MotorD	phase B- winding connection
	Pin #5 — Case/shield	(optional) shield connection

> Shield connections to the motor are not required but are recommended. Not connecting the shield signal may result in increased EMI (electromagnetic interference), reduced immunity to ESD (electro static discharge), or electrical noise resulting in motor operation failure.

2.2.3   Communication Connections

For this quick start installation we will set up the DK58113 board for serial RS232 communications. To set up the board for operation in other communication modes, see Section 4.2, "Communication Ports."

A serial port accessory cable is included with the DK58113. This serial cable (PMD p/n Cable-USB-DB9) should be connected to the DK58113 board's J5 Serial Connector, while the opposite end of the serial cable should be connected to one of your computer's USB ports.

2.2.4   Power Connections

The following table summarizes the power connection from the DC power supply to the DK58113 board. The HV voltage is the voltage at which the motor is driven and must be in the range of 12V - 56V. Board logic power is derived from this same power input using an on board DC-DC converter.

All connections are made through J1, the HV Power Connector, which is a Phoenix Contact 2-circuit terminal block Connector.

Pin #	Signal Name	Description
	J1 — HV Power Connector	
1	HV	Positive HV voltage power
2	GND	HV voltage power ground

The DC power supply recommended for use during this quick start guide is a general purpose bench-top supply with sufficient voltage and current output for the motor you expect to drive.

2.3   Step #3 — 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 DK58113's green power LED should light. This LED is locatable using Figure 2-1. If the LED does not light, recheck connections.

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

2.4   Step #4 — First Time System Verification

The first time system verification procedure summarized below has two overall goals. The first is to connect the DK58113 board with the PC that is being used so that they are communicating properly, and the second is to initialize the controlled motor and bring it under stable control capable of making trajectory moves. While there are many additional capabilities that Pro-Motion and the DK58113 board provide, these steps will create a foundation for further, successful exploration and development.

During this first time system setup you may find it useful to refer to the *Magellan Motion Control IC User Guide* to familiarize yourself with operation of the MC58113 IC, which is at the heart of the DK58113 board.

2.4.1   Establishing Communications

The first step in the first time system verification sequence is to establish serial communications, which is done as follows:

1   Make sure the DK58113 board is powered and connected to the PC via the USB to DB9 serial cable.

2   Launch the Pro-Motion application.

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

The purpose of the Interface dialog box is to indicate to Pro-Motion how your PMD controller should connect to the PC. It provides various selectable communication options such as serial and CAN.
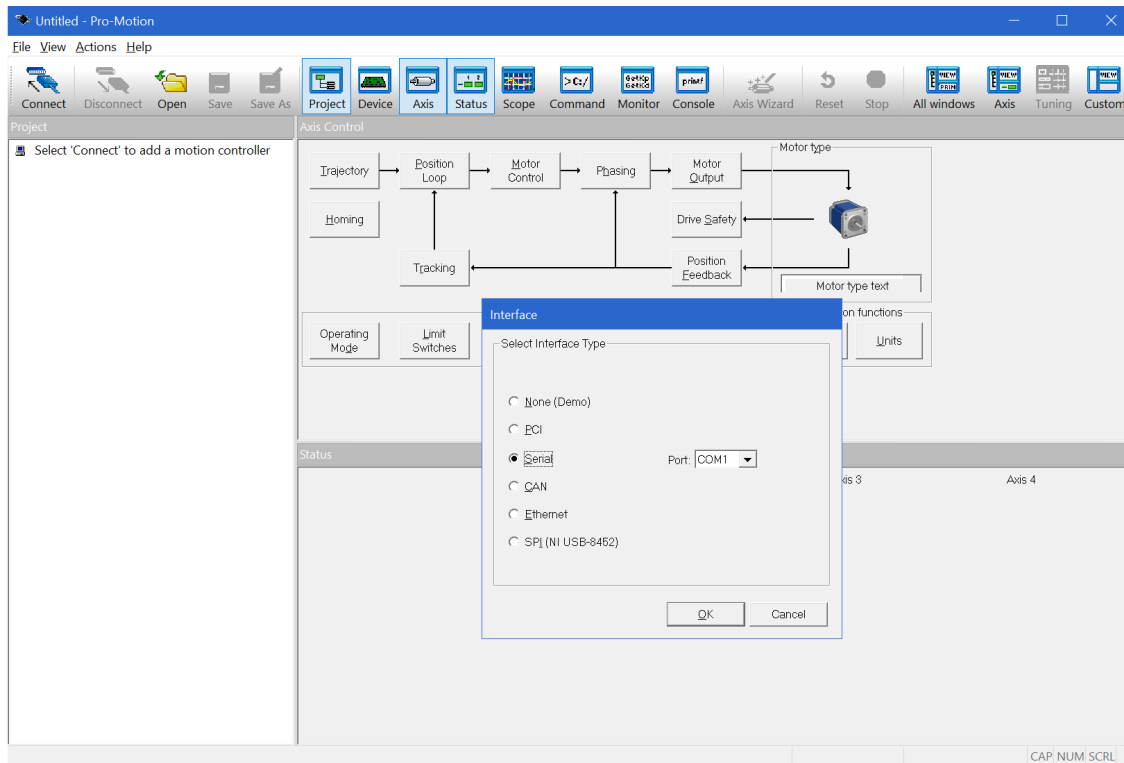
3   Click Serial and view the available COM ports listed in the Port field. If you know which COM port your USB serial cable is connected to, select it.

If you are not sure which of the listed COM ports is the correct one, you may use the following procedure:

- First, unplug the serial cable from your USB port and exit the Interface dialog box. Now re-enter the Interface dialog box by clicking "Connect" which is an icon at the far left of the top icon bar. Select Serial, and view the COM port list. Record the list of COM ports.

- Next plug the serial cable back into the USB port and once again exit and re-enter the Interface dialog box. Select Serial and now when you view the COM port list you should see a new COM port listed. This is the COM port that is connected via the serial cable provided with the DK.

- Select this COM port and hit the OK button.

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

4   Click OK without changing any of these settings.

If serial communication is correctly established, after a brief pause a set of object graphics loads into the Project window to the left, as shown in the following figure.
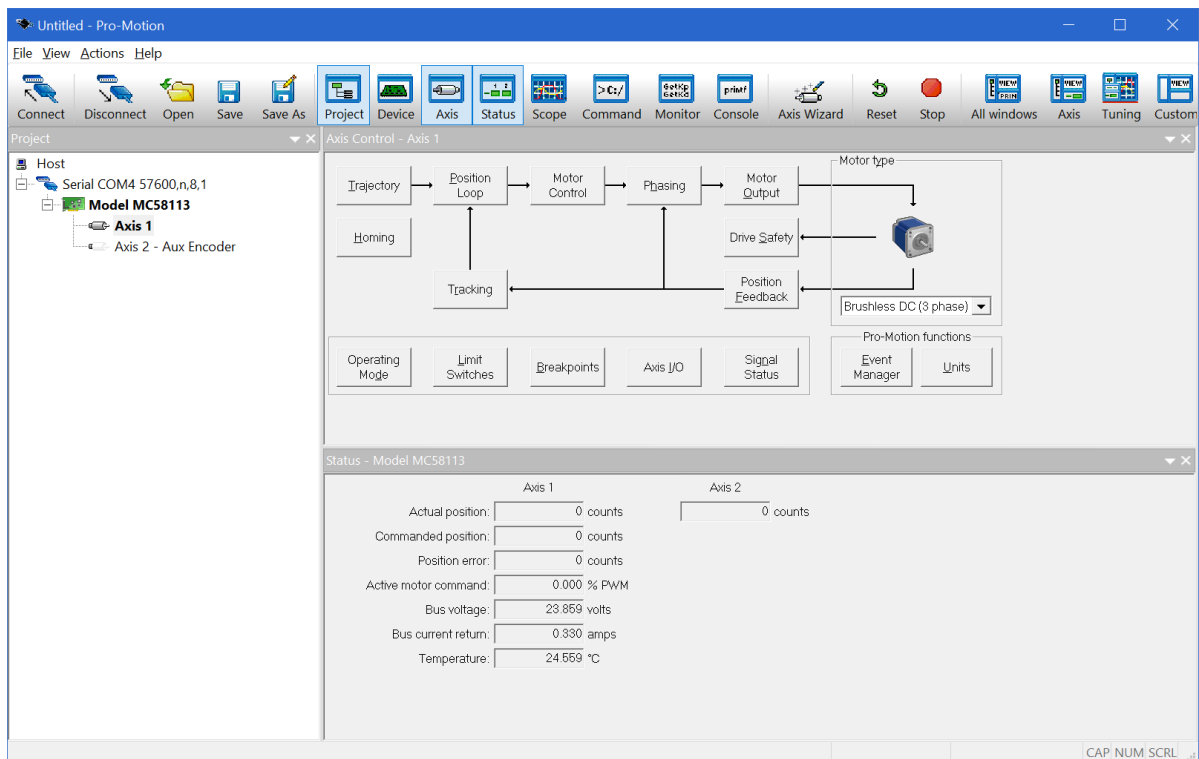
If serial communications are not correctly established, a message appears indicating that an error has occurred. If this is the case, recheck your connections and repeat from step 1.

2.4.2   Running the Axis Wizard

The next step is to initialize each axis in your system, thereby verifying correct motor connections, and other connect-tions. All of this can be conveniently accomplished using Pro-Motion's Axis Wizard.

Before selecting the Axis Wizard icon however you should select the axis # to initialize. This is accomplished via the Project window, which is located to the left, by selecting the desired axis. In the case of DK58113 boards there is only a primary axis (axis 1) and an auxiliary encoder input axis (axis 2). You should select axis 1. After clicking on the desired axis it will become highlighted and the Axis Control window title will change to reflect the newly selected axis #. If the axis is already highlighted there is no need to select the axis.

Once the desired axis to initialize has been selected you should:

1   Click the Axis Wizard toolbar button which is located right of center in the row of icons toward the top of the window.

The Axis Wizard initialization window appears.

2 Click Next and follow the Axis Wizard instructions for each page of the axis setup process. A typical axis wizard sequence takes 5-10 minutes depending on the motor type and number of motion peripherals such as limit switches your system uses.

Toward the beginning of the Axis Wizard screen sequence you will be given the option to select 'DK58113' as the hardware platform being used. Make sure to select this option because it sets important control and safety values for the board's on-board switching amplifier. If you are using another DK58113 family product such as DK51113, DK53113, or DK54113 you should still select the DK58113 option.

Although rare, if you have any problems while going through the Axis Wizard you may find it helpful to view Section 3.12, "Troubleshooting Suggestions." Another section that you may find helpful is Section 3.11.2, "Application Note — Tuning the Position Loop."

3   The last Axis wizard screen allows you to save the various control parameters you have specified while in the Axis Wizard. This is convenient because it allows you to quickly load your control settings and make them active without the need to re-run the Axis Wizard.
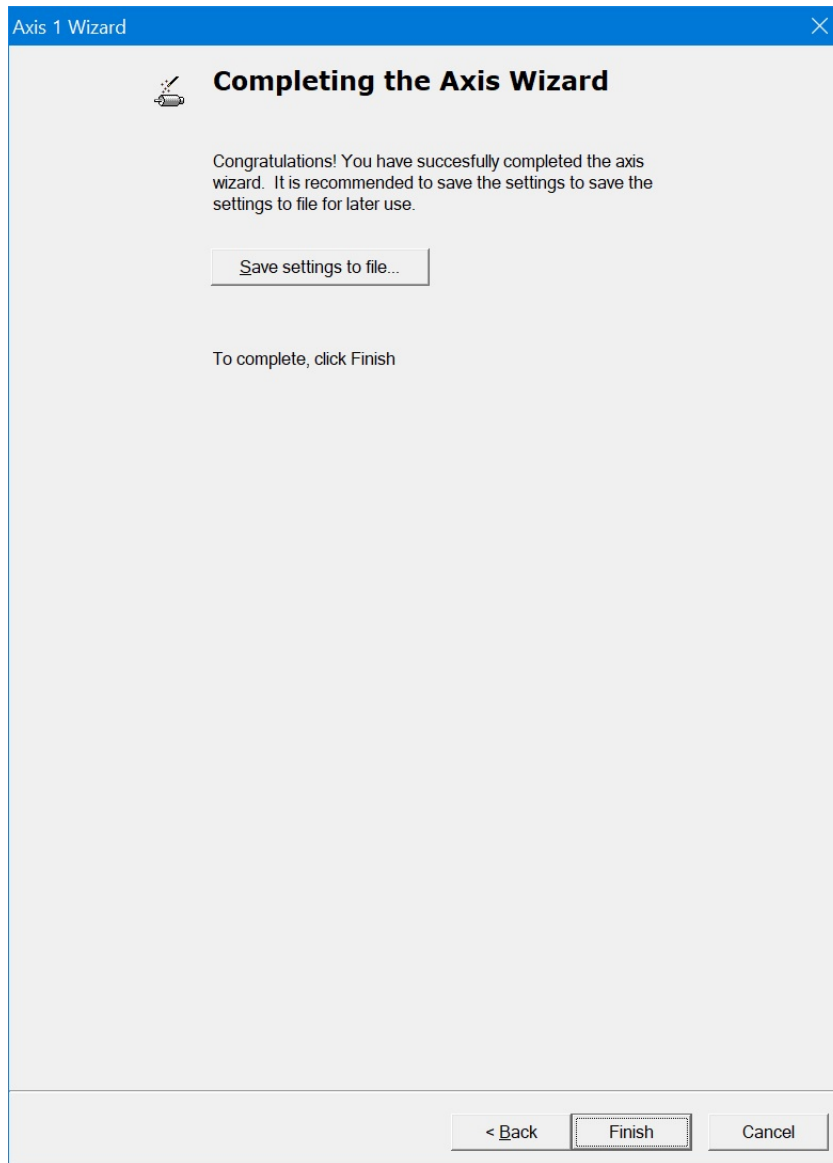
4   To save the Axis Wizard settings select "Save settings to file…" and specify a destination directory and file name. The Axis Wizard will create the file with your parameters loaded into it. For more information on saving and restoring configuration settings see Section 3.9, "Project Configuration Save & Restore."

5   When you have specified a file name and saved your settings select Finish at the bottom of the screen. You will now be returned to the main Pro-Motion screen.

2.4.3   Exercising The Motor

The next step is to exercise the motor and verify that it is reacting correctly to programmed commands.

This is accomplished by having the motor make a simple point-to-point move. To perform a point-to-point move:

1   Click the Trajectory button in the Axis Control window. The Trajectory dialog box appears, an example of which is shown below.
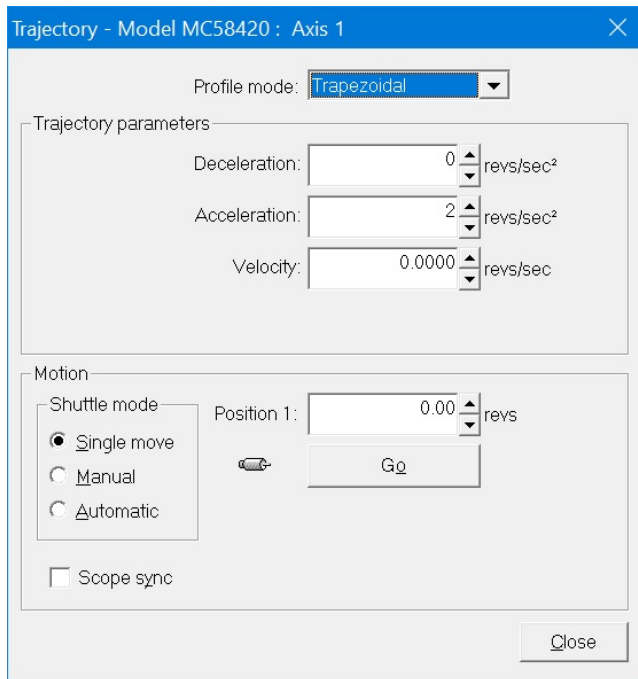
2   In the Profile mode list, select Trapezoidal, and in the Motion box select Single move.

3   Enter motion profile settings for deceleration, acceleration, velocity, and destination position (Position 1) that are safe for your system and will demonstrate proper motion. The units of these parameters should match the units you selected earlier in the Axis Wizard setup process. If you would like to change the units you can do this by going to the Axis Control window and clicking the Units box which is to the far lower right. You should then exit and re-enter the Trajectory dialog box for the units change to be visible.

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

Congratulations! Quick setup for your DK58113 is now complete.

2.5   Next Steps

When ready, to continue exercising and optimizing the motion system and to begin developing the software for your control application refer to the subsequent chapters of this manual listed below.

Chapter 3, *Going Further with Pro-Motion*, shows you the most frequently used features of Pro-Motion. You will find the content in this chapter helpful if you are new to Pro-Motion and PMD's Magellan Motion Control ICs.

Chapter 4, *DK58113 Board Operation*, provides a complete reference for operation of the board.

Chapter 5, *Communication Port Connections*, shows you how to set up alternate communication channels for the PMD controller you installed in this quick setup guide.

Chapter 6, *Software Development*, provides an overview of how to develop software application code to control your PMD-based control system.

This page intentionally left blank.

3. Going Further with Pro-Motion

In This Chapter

▶ Pro-Motion Screen Layout

▶ Project Window

▶ Device Control Window

▶ Axis Control Window

▶ Status Window

▶ Monitor Window

▶ Command Window

▶ Scope Window

▶ Project Configuration Save & Restore

▶ Configuration Export to C-Motion

▶ Pro-Motion Application Notes

▶ Troubleshooting Suggestions

In this chapter we provide more information on Pro-Motion to help familiarize you with its most commonly used features.

3.1   Pro-Motion Screen Layout
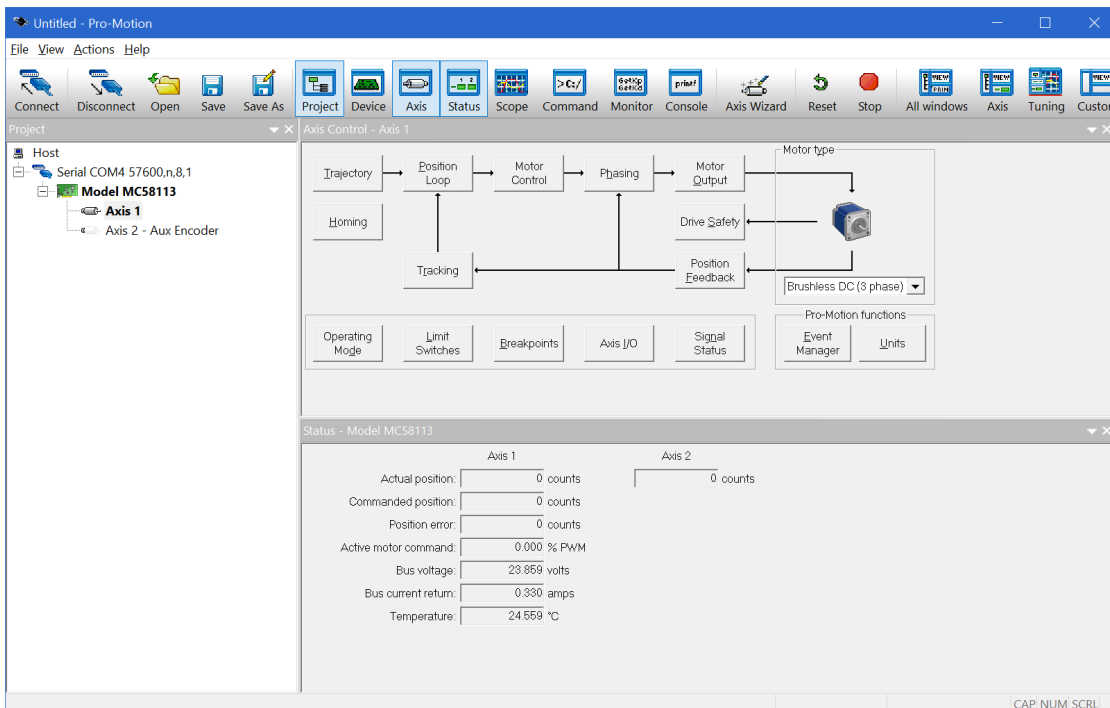
The screen capture above shows an example of Pro-Motion as it appears after first launching and connecting to a single-axis PMD controller. For information on connecting to a PMD controller after launching Pro-Motion see Section 2.4.1, "Establishing Communications."

Pro-Motion follows the general form of Windows-based applications with a menu at the very top of the application window. In the case of Pro-Motion the available menu functions are *File, View, Actions*, and *Help*. In addition there is a tool bar with icons allowing the user to access the most commonly used Pro-Motion functions with a single click. Here is a description of these clickable icons in the order that they appear from left to right and grouped by function:

Connect, Disconnect

These toolbar icons let you connect or disconnect to PMD controllers in your motion hardware setup.
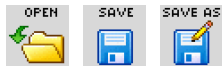
Open, Save, Save As

These toolbar icons let you call up and save your setup configuration. Section 3.9, "Project Configuration Save & Restore," describes these "project file' mechanisms in more detail.
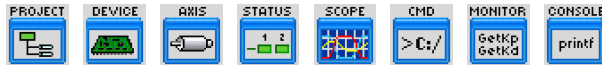
Project, Device Control, Axis Control, Status, Scope, Command, Monitor, and CME Console

Each member in this group of icons represents a Pro-Motion window that, when clicked, opens if not yet being displayed or closes if being displayed. We will discuss the functions provided by many of these windows later on.

Axis Wizard

This icon starts the Axis Wizard. The Axis Wizard is the recommended way to establish and verify connections between the PMD controller and each axis of the motion hardware setup. See Section 2.4.2, "Running the Axis Wizard," for more on the Axis Wizard.

Reset, Stop

The Reset button will immediately reset the currently selected device in the Project window. The Stop button will immediately stop the motion of the axis selected in the Project window.
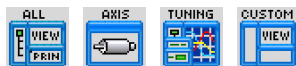
All, Axis, Tuning, Custom

This group of icons controls how the Pro-Motion windows are arranged. They provide convenient pre-programmed arrangements of windows designed to make specific tasks easier. The default arrangement is Axis.

Windows can also be arranged manually and saved as a custom window arrangement. To store the arrangement of windows in your Pro-Motion session use the *View/Save Custom View* menu function at the very top of the Pro-Motion screen. Thereafter, selecting the Custom icon will present the windows in this saved custom view scheme.

3.2 Project Window
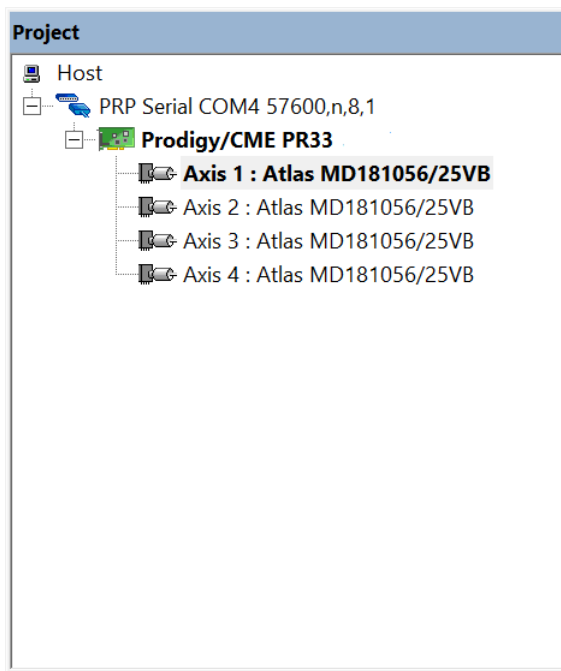
The Project window shows all PMD controllers presently connected to Pro-Motion. It presents these connections using a Windows Tree View scheme. The above screen capture shows an example Project window for a four axis controller.

In addition to displaying an icon and part number for each connected PMD controller the Project window displays a 'motor' icon for each axis within that controller. So a Prodigy board with four active axes would have four such motor icons displayed. Single axis devices such as a DK58113 board or N-Series ION drive would have two such icons displayed, one for the primary and one for the auxiliary encoder input. For Magellan axes that use an Atlas amplifier a special version of the motor icon is displayed and the part number of the attached Atlas is displayed next to the motor icon.

In addition to displaying the addressable PMD devices in the present Pro-Motion session, the Project window is used to select which axes are actively being processed by Pro-Motion windows. For example in the screen capture above, axis #1 is highlighted, which means axis-specific windows such as the Axis Control window will program the settings for axis #1. To change the currently selected axis simply click on the desired axis icon in the Project window.

3.2.1   Device & Axis Label Customization

A very useful feature of the Project window is that the device and axis labels generated automatically by Pro-Motion can be customized to suit the application. This is shown in the screen capture below, where labels from the previous example Project window have been changed.
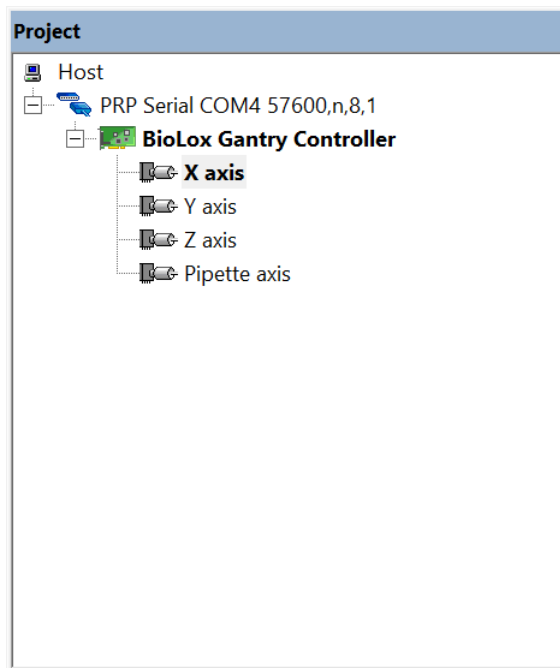
To change the device or axis labels slowly click twice on the existing text, and then type in your desired new entry.

3.3   Device Control Window
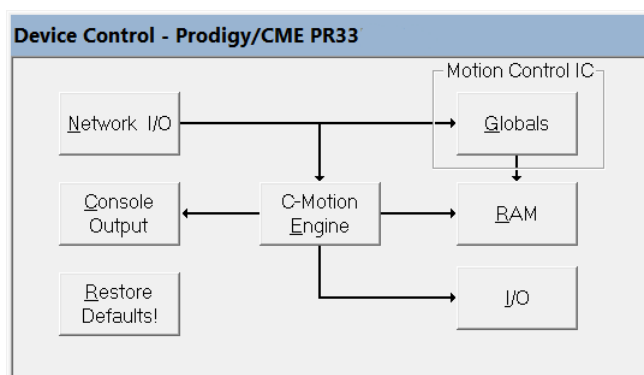
The Device Control window allows you to view and update 'device level' parameters. Devices are PMD controllers such as motion IC developer kit boards, Prodigy boards, and ION Drives. Different products have different device-level functions available and the Device Control window displays the available functions for the selected device as clickable boxes. To select a particular device from the Project window click any axis controlled by that device.

The list below briefly describes the clickable function boxes you may see in the Device Control window:

Network I/O – The Network I/O module of the Device Control window allows you to view and set various parameters for each of the communication ports supported by the connected-to device.

Motion Control IC Globals - Clicking this box lets you set the Magellan or Juno IC's cycle time and lets you view various characteristics of the motion IC such as the IC family name, supported motor type(s), number of supported axes, and the version #.

NVRAM – Some Motion Control ICs support non-volatile memory which can contain initialization command sequences. For those products this function allows you to view, erase, and download script file content or the current Pro-Motion configuration to the Magellan IC's NVRAM.

RAM – Allows you to view the size of the PMD controller's RAM used for motion trace and User Defined Profile Mode storage. You can also change the usable size of the RAM with this function.

Analog Input – Allows you to view current reading(s) from the PMD controller's general purpose analog input function.

C-Motion Engine – This function allows you to view, erase, and download a *.bin* user code memory image to the C-Motion Engine. In addition this function allows you to manually reset, start, or stop code execution, and set whether user code begins execution automatically upon power-up or only after manual start.

Console – This function lets you specify the console communication channel and parameters used in connection with user code running on the C-Motion Engine.

Restore Defaults – This function reverts the PMD controller's parameters to their default conditions. Any changes that may occur as a result of this operation occur in the device's NVRAM. Only after a reset or power cycle will NVRAM values become the active settings used by the PMD controller. Particular care should be taken when using this function if communication parameters have previously been altered because restoring them to their default values may mean that Pro-Motion no longer uses the correct settings to communicate with the PMD controller. For a detailed list of NVRAM-stored defaults that may be affected by this operation refer to the **DeviceSetDefault** command description in the user manual for the product you are using.

Digital I/O – This module allows device-level digital I/O registers to be read and set by the user.

The list of clickable function boxes above represents a superset of the boxes that may appear is the Device Control window for a particular PMD controller. The actual boxes displayed is based on the type of PMD controller that is selected. For example, motion IC developer kit boards such as DK58113 or DK58420 will display fewer selectable boxes while products such as ION/CME Digital Drives and Prodigy/CME boards will display more of these boxes.

3.4   Axis Control Window

The Axis Control window allows you to view and update motion control parameters for the axis selected in the Project window. Each selectable box (technically these are Windows buttons) in the Axis Control window results in a dialog box being opened letting you access a sub-set of the motion control functions provided.
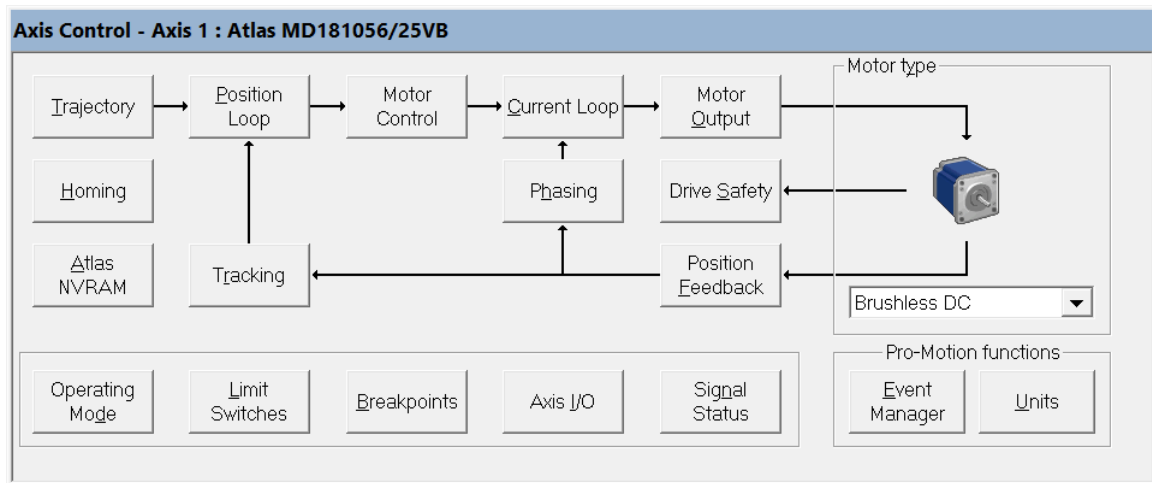
The Axis Control window presents these selectable boxes such that the overall control flow for the motor type selected is evident. For example the boxes and control flow arrows displayed for a Brushless DC motor are different than for a step motor reflecting the fact that Brushless motors are controlled differently than step motors.

Some of the boxes at the bottom and left hand side of the Axis Control window are not connected via the control flow arrows. These provide access to motion control settings such as for limit switches, breakpoints, axis I/O, signal status, event management, units, homing, and Atlas NVRAM (for axes which use an Atlas amplifier).

The box labeled 'Operating Mode' provides control of whether major axis control modules are active. These modules are trajectory, position loop, current loop, and motor output. While normally enabled, there may be circumstances, for example if a motion error occurs, where some modules will get disabled for safety reasons and may need to be manually re-enabled.

Underpinning the control flow arrows, selectable boxes, and Operating Mode status in the Axis Control window is the control architecture of PMD's Magellan Motion Control IC, which is the motion controller at the heart of all PMD position control products including Magellan Motion Control IC developer kit boards, Prodigy boards, and ION drives. The reference manual that describes this is the *Magellan Motion Control IC User Guide*. For example if you want to know what trajectory profile modes are available and exactly how they function this manual contains that information. The same applies for the other selectable boxes and associated functions within the Axis Control window.

3.5 Status Window

The screen capture below shows an example of a Status window display, in this case for a four axis controller. The Status window displays all axes for the device selected in the Project window. For example if one of the axes in a four-axis Prodigy/CME Machine-Controller is selected all four axes of that controller will display in the Status window.

Status - Prodigy/CME PR33				
	Axis 1	Axis 2	Axis 3	Axis 4
Actual position:	0.000 revs	0 revs	0 mm	0 mm
Commanded position:	0.000 revs	0 revs	0 mm	0 mm
Position error:	0.000 revs	0 revs	0 mm	0 mm
Active motor command:	0.000 % PWM	0.000 % PWM	0.000 % PWM	0.000 % PWM
I²t energy:	0.000000 amps²sec	0.000000 amps²sec	0.000000 amps²sec	0.000000 amps²sec
Bus voltage:	24.767 volts	24.868 volts	24.860 volts	24.599 volts
Temperature:	32.465 ℃	32.414 ℃	33.703 ℃	33.730 ℃

For each axis of the selected PMD controller the value of several parameters are continuously displayed in the Status window. Note that some PMD products, particularly those without an amplifier, will not display all of these parameters. Here are brief descriptions of the displayed parameters:

Actual position – This is the actual position of the motor as measured by an encoder or by Hall sensors if Halls are programmed to provide position feedback.

Commanded position – This is the instantaneous commanded position from the trajectory generator.

Position Error – This parameter shows the difference between the commanded and the actual position. For servo-controlled motors (DC Brush and Brushless DC) this value measures how accurately the position loop is maintaining the commanded position. For step motors, if an encoder is present, this represents the amount that the actual motor lags the commanded position. A positive value means the commanded position is greater than the actual position, and vice versa for a negative value.

Active motor command – This parameter shows the torque (current) command being sent to the current loop/amplifier. If there is no current loop present (or active), rather than units of amps this parameter will have units of % PWM (Pulse Width Modulation).

Bus voltage – For PMD products which include an amplifier function this parameter indicates the current measured value of the DC supply voltage.

Bus current return – For PMD products which include an amplifier function this parameter continually indicates the current flow from HV to ground in the amplifier's switching bridge. Note that this current value does not include current used to power internal logic of the PMD controller.

Temperature – For PMD products which include an amplifier function this parameter indicates the instantaneous temperature of the amplifier bridge.

The quantities displayed in the Status window are displayed using the user-selected units. For example in the screen capture above axes 1 and 2 use position units of revolutions, and axes 3 and 4 use units of millimeters. To change Pro-Motion display units for a particular axis select the 'Units' box in the Axis Control window.

3.5.1 Status Polling

By default Pro-Motion continually polls the device selected in the Project window so that the Status window shows up-to-date values for all axes of that device. In addition, regardless of what device is selected in the Project window, Pro-Motion regularly polls all connected devices to report on safety events such as motion error, overcurrent, etc…

By default this background status polling is enabled, however there are situations where it may become disabled. The first is after some types of communication errors between Pro-Motion and the connected PMD controller. To avoid continually generating communication error warnings automatic polling may be disabled by Pro-Motion. If this happens Pro-Motion will display this change in a dialog box that pops up after the communication error.

Automatic polling may also be manually disabled by the user. This can be accomplished via the *View/Status Polling* menu function. The main reason for manually disabling background polling is when using the Monitor window. Background polling results in a large number of messages being displayed in the Monitor window, thereby making it difficult to focus on specific transactions of interest. Disabling automatic polling solves this problem. Note that after status polling is disabled it should eventually be re-enabled to restore safety monitoring.

> Disabling status polling may result in erroneous or unsafe operation of the motion controller being unreported by Pro-Motion. It is therefore always recommended that Pro-Motion status polling be enabled during regular operation of the motion controller.

3.6 Monitor Window

```
Monitor - Prodigy/CME PR33
SetJerk 0
SetPosition 123456
Update
ResetEventStatus 0xffef
ResetEventStatus 0xffef
ClearPositionError
Update
ResetEventStatus 0xffef
SetSettleTime 0
SetSettleWindow 0
SetTrackingWindow 0
SetMotionCompleteMode 0
SetPositionErrorLimit 0
SetEventAction EventMotionError(3) 0x0005
ClearPositionError
Update
ResetEventStatus 0x8000
RestoreOperatingMode
ResetEventStatus 0x8000
RestoreOperatingMode
SetProfileMode 0
SetStartVelocity 0
SetAcceleration 43
SetDeceleration 21
SetVelocity 419430
SetJerk 0
SetPosition 0
Update
```

Pro-Motion supports the ability to view communications between Pro-Motion and a connected PMD controller. These communications are in the form of hexadecimal-coded packets encoding short command-oriented transactions sent by the PC and responded to by the PMD controller. Packets can transmit messages which have meanings such as "Set the Profile Destination Position to 123,456" or "What is the current encoder position?" Transmitted command packets are displayed in the monitor as ASCII mnemonics. For example these two command functions are displayed in the monitor with the mnemonics **SetPosition** and **GetActualPosition**.

While many users will not need to concern themselves with the format of command mnemonics, software developers in particular may find the Monitor window traffic useful to learn how Pro-Motion commands the PMD controller to achieve certain functions.

Here is some additional information about the Monitor window that you may find helpful:

- To open the Monitor window click on the corresponding icon on the top menu bar.

- Displayed commands have zero, one, two, or three arguments. For details on command and argument formats refer to the *C-Motion Magellan Programming Reference*.

- Arguments that are prefaced with a "0x" (for example 0x1234) are being displayed in hexadecimal. Arguments without an 0x (for example 1234) are being displayed in decimal.

- To control which command packets are displayed right-click from within the Monitor window and select one or both of the 'filter' settings. *Filter Gets* will avoid displaying traffic due to regular background queries from Pro-Motion to the device. Note that an alternative to disabling all Get commands may be to disable automatic status polling. See Section 3.5.1, "Status Polling," for information on this. *Filter Reads* will avoid displaying traffic containing scope trace data.

- You can clear the content of the Monitor window using right-click and *Clear*.

- You can save the content of the Monitor window to a text file using right-click and *Save As...*

> The command packets that are displayed in the Monitor window consist only of Magellan Motion Control IC packets. If the PMD controller is a PRP Device, MotionProcessor (Magellan) command packets will be displayed but packets associated with Device, Peripheral, Memory, or CMotionEngine resource command traffic will not.

3.7 Command Window

```
Command - Prodigy/CME PR33
For a list of commands press Tab
 > #Axis 1
 > #Axis 1
 > #Axis 1
 > GetVersion
0x58420031
 > SetProfileMode 0
 > SetStartVelociny
Error, syntax error in command string.
 > SetStartVelocity 12345
 > SetAcceleration 43
 > GetAcceleration
0x0000002b 43
 > SetVelocity 5000
 > GetVelocity
0x00001388 5000
 > SetPosition 123456
 > GetPosition
0x0001e240 123456
 > ClearPositionError
 > SetSettleWindow 50
 > SetSettleTime 100
 > SetTrackingWindow 123
 > Update
 >
```

Pro-Motion supports a command line interface known as the Command window that allows the user to directly type in and send commands to the attached PMD controller.

The Command window has a DOS command line style interface with a ">" command prompt. All Magellan Motion Control IC commands are accepted, and as described in <u>Section 3.6, "Monitor Window,"</u> are expected to be in ASCII mnemonic format.

The *C-Motion Magellan Programming Reference* provides detailed syntax and argument definitions for each enterable command. Go to the alphabetized Instruction Reference section of this manual to lookup specific commands. The command syntax is indicated at the top with argument definitions provided just below. Note however that the *Axis* argument, which is shown as the first argument, should not be entered when typed into the Command window in the command line. Doing so will result in an error indicating there are too many arguments. As indicated above the addressed axis is specified using a separate #Axis command.

Here is some additional information on the Command window that you may find helpful:

- To open the Command window click on the corresponding icon on the top menu bar.

- Both the command mnemonic and associated argument values are entered on a single line followed by the Enter key. If the expected number of arguments are not entered an error message will display.

- Command mnemonics are not case sensitive.

- Entered arguments are separated from the command mnemonic and from each other (if the command accepts more than one argument) by spaces.

- All arguments are numerical. Arguments can be provided in decimal format (for example 1234) or in hexadecimal format by prefacing with "0x" (for example 0x1234).

- A facility for viewing the list of available commands is activated by hitting the Tab key. As characters are typed in, whenever the Tab key is pressed the command mnemonics that match the characters typed in are displayed in a "Select a Command" window that pops up. If Tab is pressed at the prompt a list of all available commands is displayed. To select a command from the list highlight the command and press Enter. The selected command appears at the command prompt (>) and the 'Select a Command' window closes.

- Upon entering the Command window typed-in commands are sent to the axis presently selected by the Project window. This can be changed while inside the Command window by entering a "#" followed without a space by "Axis n" (n being the axis number to change to) at the command prompt. For example after entering the command line "#Axis 3" all subsequent typed-in commands will apply to axis 3 of the PMD controller. The addressed axis can be changed by further #Axis commands, or by exiting the Command window, at which point the active Pro-Motion axis returns to the axis selected in the Project window.

- If there is an error in processing a typed-in command a message will display indicating the nature of the error. An example of this in the form of a "Error, syntax error in command string" message can be seen in the Command window screen capture above. Commands that request information will return the requested value on the next command line. Commands that do not request information will display the command prompt at the next line.

3.7.1   Command Window Scripts

```
Command - Prodigy/CME PR33                                    ✕
'
 > '----------------------------------------
 > ' Example script
 > ' This script loads S-curve profile parameters
 > ' and Servo gains into the motion IC
 > '----------------------------------------
 > '
 > SetProfileMode 0            ' Set to trapezoidal profile
 > SetStartVelocity 0          ' Set start velocity value
 > SetVelocity 10000           ' Set velocity value
 > SetAcceleration 500000      ' Set acceleration value
 > SetDeceleration 800000      ' Set deceleration value
 > SetJerk 1234567             ' Set jerk value
 > SetPosition 123456          ' Set destination position
 > '
 > ' Now set some position loop gain parameters
 > '
 > SetPositionLoop 0 123       ' Set Kp Proportional gain
 > SetPositionLoop 1 25        ' Set Ki Integrator gain
 > SetPositionLoop 2 10000000  ' Set Ilim Integrator limit
 > SetPositionLoop 3 4560      ' Set Kd Derivative gain
 > SetPositionLoop 4 1         ' Set Dtime Derivative time
 > SetPositionLoop 5 32767     ' Set KOut Output gain
 > '
 > ' Get a few values
 > '
 > GetPositionLoop 0           'Get Kp
0x0000007b 123
 > GetPosition                 'Get Destination position
0x0001e240 123456
 >
```

In addition to the features described above the Command window supports a very useful feature which is the ability to execute a series of commands stored in a script file. Command window scripts do not support conditional statements or branching, instead they act more as macros executing a fixed sequence of pre-programmed commands. As such they are generally used just for sending parameter initialization sequences.

Command window script files are text files containing only ASCII characters. Within Windows the Notepad program is a common way to edit text files, but any editor that supports a text file format can be used.

From a Command window session, to execute a script the "<" character is entered followed without a space by the script file name, for example *<c:\scripts\TestScript.txt*. Alternatively entering just the "<" character following by the Enter key will call up a Windows Open dialog box. An alternate way to access this same dialog box is right-clicking from the Command window and selecting *Specify Script File*.

Here is information on the formatting of script text files:

File type – Script files are ASCII text files

Comments – Comments are content embedded in the script file that are not executed, but rather are used by the script developer to clarify its content. Comments may appear anywhere within a line and result in all subsequent characters to the end of the line being ignored during script processing. The character recognized as beginning a comment is the single quote '.

Blank lines, leading spaces and tabs – Blank lines are ignored as are leading spaces or tabs.

#Axis commands – Scripts may contain #Axis commands anywhere in the script. This is the mechanism by which a script can send commands to different axes on a device that supports multiple axes.

Commands and Newlines – Only one command and its associated arguments (if any are needed) are recognized per line. Recognized characters marking the end of the line are \R (ASCII 13) and \N (ASCII 10).

Argument delimiters – Arguments must be separated from adjacent commands or arguments by either a space or a tab. Commas are not an allowed delimiter.

Here is some additional information on Command window scripts that you may find helpful:

- If during script processing an error is encountered script processing will halt.

- Nesting of scripts is allowed, meaning scripts are allowed to contain command lines that execute other scripts. If using this feature caution should be exercised to not have a script call itself.

- The primary intended purpose of Command window scripts is to load Magellan Motion Control IC parameter settings so that they don't have to be typed in manually. While not specifically prohibited, script commands that result in axis motion being initiated or modified is not recommended and should instead be done via the Pro-Motion Axis Control window, or via user-written application code.

3.7.2   Motion IC Settings Export to Script

Pro-Motion provides the ability to export the Magellan Motion Control IC settings for a selected device to a Command window script.

The configuration information that is saved to a script file consists of Magellan IC settings such as position loop gains, safety settings, signal sense, etc. Multi-axis products save settings for each axis. For example a four-axis Prodigy/CME Machine-Controller would have the configuration information saved for all four axes.

Here is some additional information that you may find helpful:

- To save the configuration to a Command window script file use the menu function *File/Export Motion IC Settings as Script*. A default file name is provided which can be changed. The specified script file will be created and displayed in Windows Notepad for convenience.

- Saved configuration script files are compatible with the Command window script facility and can be loaded and executed from the Command window. In addition the content can be edited or copied and incorporated into other script files as desired.

- Configuration scripts are ASCII-formatted files and therefore cannot be sent directly to the PMD controller using a standalone terminal or PC-based terminal emulator such as Procomm.exe. Scripts are parsed by the Command window and converted into hexadecimal-coded packets, which are then sent to the PMD controller.

- For script file exports there are generally more parameters saved than are actually used in a given application. If desired, the user can therefore review the saved configuration data and remove unused setting entries.

> Only Magellan Motion Control IC configuration settings are exported to scripts via this export function. Non MotionProcessor resource configuration settings are not exported as part of this function.

3.7.3   Downloading Scripts to the Motion IC NVRAM

MC58113-series ICs provide the ability to store configuration information such as communication settings, gain parameters, drive-related safety parameters, and other parameters into the internal NVRAM (non-volatile memory) of the motion IC. This initialization configuration content is specified using Pro-Motion Command window scripts.

Once loaded into the MC58113's NVRAM, the commands in the specified script are automatically executed and any parameter settings contained in the script become the active settings used by the IC at power-up or reset. The main advantage of storing initialization sequences in NVRAM is that the stored settings are available immediately after powerup of the IC, rather than having to be downloaded by a host PC or microcontroller.

Programming or re-programming of the motion IC's NVRAM is accomplished by selecting the NVRAM box in Pro-Motion's Device Control window. The screen capture below shows the dialog box that appears.
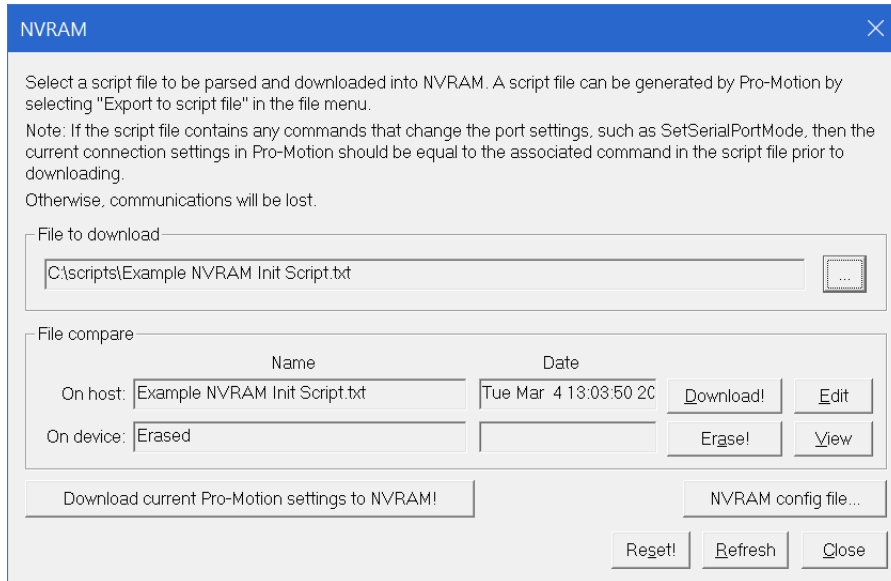
Specify the script file in the "File to download" field. If you click the small box to the right a Windows Open dialog box will appear. When ready click Download! to store the script file content into the IC's NVRAM. Similar to scripts that are executed in the Command window and sent to the IC via a communication port, the script content is parsed by Pro-Motion, converted into hexadecimal-coded packets, and then sent to the motion IC for storage.

If the NVRAM already contains stored content it will be overwritten when a new script is downloaded. To erase the IC's NVRAM memory select the Erase! Button. This will restore the motion IC to use its default parameter settings after powerup.

3.8   Scope Window

Pro-Motion's Scope window ties directly to the Magellan Motion Control IC feature called trace, which provides the ability to capture and store in hardware memory up to four motion registers simultaneously, at up to 20 ksamples/second (depending on the PMD controller).

Once a trace operation is specified the Magellan IC captures the motion values at a programmed time interval and stores these values in its local RAM memory. Pro-Motion then sends commands to retrieve the data from the Magellan IC and display this data graphically. In addition to being displayed, traced data can be captured to a file for import to spreadsheets or other graphing and analysis software. This is accomplished using the *File/Export Trace* menu function.

Common traced variables include the motor output command, position error, commanded position, commanded velocity, actual position, and others. In total there are over 100 different selectable trace variables.

To access the Pro-Motion scope function click the icon at the top bar labeled *Scope*. An example screen capture of the Scope window is shown below.
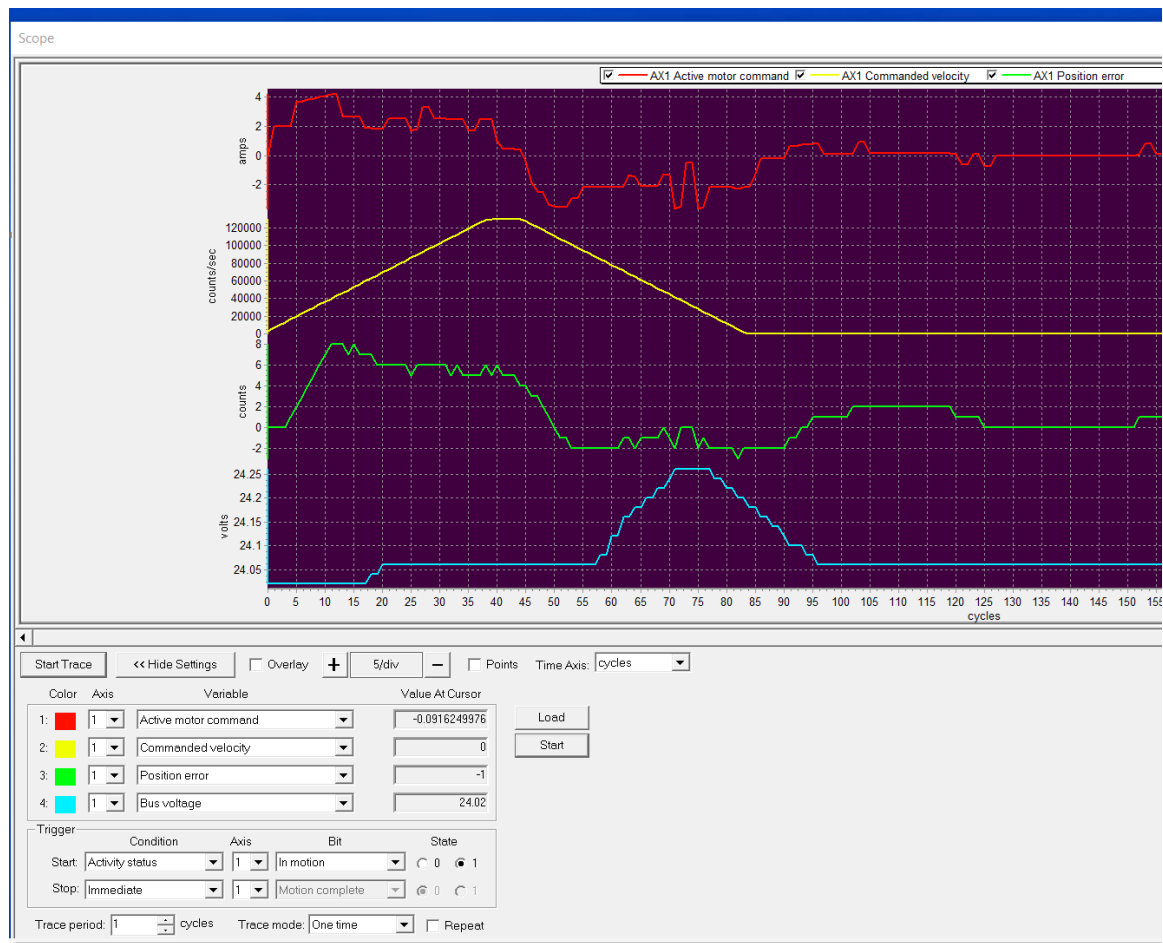
Here are some of the key settable fields of the Scope window:

Trace Variables – The core of the trace and scope function is the list of motion registers that will be traced and graphed. These are shown as Variables 1 through 4 with graph colors red, yellow, green, and blue respectively. For each trace variable click the down arrow which in turn displays a list of trace categories such as Commanded Trajectory, Feedback, Position Loop, etc... Selecting one of these categories then shows the specific available traceable motion variables. Note that for multi-axis devices variables can be traced from separate axes.

Time Axis – The top portion of the Scope window graphs captured data. Up to four variables can be graphed at the same time. The horizontal scale is time, with selectable units via the Time Axis field of cycles, milliseconds, and seconds.

Trigger Controls – Similar to a regular oscilloscope the conditions by which trace data collection can start or end is settable. Use the down arrow key to see the list of available trigger registers, and the associated bit or signal for each register. For examples of setting trigger controls see Section 3.11.1, "Application Note — Determining the Optimum Trajectory Acceleration," Section 3.11.2, "Application Note — Tuning the Position Loop," and Section 3.11.3, "Application Note — Determining Acceleration Feedforward Gain.".

Trace Period & Trace Mode – The data capture trace period is expressed in cycles, which are 51.2 µSec per cycle. So specifying a period of 1 cycle captures data at ~20 kHz. Trace mode can be set to *One-time* or *Rolling Buffer*. One-time capture fills the trace buffer only once beginning when the trigger start condition is satisfied. One time capture is recommended unless a particular reason for selecting rolling buffer exists.

The most common use of rolling buffer mode is to display trace data leading up to the moment an event occurs. For example with rolling buffer mode selected, if *Immediate* is set as the trigger start condition and the *Event Status* register selected with *Motion Error* as the programmed bit and the state set to 1, after a motion error occurs the captured and displayed data will show the trace data prior to the occurrence of the motion error. For more information on motion errors refer to the *Magellan Motion Control IC User Guide.*.

> When using rolling buffer mode caution should be exercised to avoid overflowing the buffer, which can result in the graphed data not matching the actual traced data. This can occur if the rate of data being put into the trace buffer by the Magellan IC is faster then the rate at which Pro-Motion can request it.

Trace Length – The total number of trace samples can be specified via this setting. Setting this value may help you better-manage how your data displays or how it exports to a file. The programmed value represents the number of data set captures. For example if this value is set to 500 and one variable is traced a total of 500 data values will be captured, if two variables are traced a total of 1,000 data values will be captured etc...

3.8.1   Trace Buffer Display Optimization

The speed of the scope display update is related to the size of the buffer that fills with data in the Magellan IC, and the speed with which the Windows PC can retrieve this data from the Magellan IC. For higher speed interfaces such as CAN or Ethernet optimizing these settings is usually not needed. However for serial interfaces, improving the communication speed may be useful. To set the baud rate to a new value see Section 5.1.2, "Changing the Active RS232 Settings."

Another way to speed up serial communications is to reduce the latency between message packet sends. This is done via the following sequence; First, open the Windows Device Manager by typing "Device Manager" in the Windows search box. Next, find Ports (COM & LPT) from the list and click on it, then right-click the USB Serial Port (COMn) of the serial port you are using and then select Properties. Click on the Port Settings tab and then select the Advanced button. Change the field for Latency Timer (msec) to a value of 1. Click OK on both Windows and close the Device Manager.

3.9   Project Configuration Save & Restore

Pro-Motion project files allow you to store a motion control configuration to a named file and later recall this saved configuration. You can save a project at any time while working with Pro-Motion by specifying *File/Save Project* or *File/ Save Project* As from the Pro-Motion session menu.

Project files save the control parameters and communication settings for all axes of all PMD controllers connected to Pro-Motion and displayed in the Project window. The files that Pro-Motion uses to save and restore project configuration information have an extension of *.PMD*. These project files are not user-readable. They encode the project configuration information in a format that is written and read by Pro-Motion, but is not intended to be read or edited by the user.

The specific configuration information that is saved via the *File/Save Project* function is all of the PMD controller's configuration, control, and communication settings including all Magellan Motion Control IC settings. The main exceptions are content that is stored in NVRAM such as Magellan initialization commands, device SetDefaults settings, and *.bin* C-Motion Engine user code programs. Most of the Pro-Motion windows also have their content saved via the project save mechanism. This includes the Trajectory and Units dialog box settings of the Axis Control

window, and the Scope window settings. Finally, the operating mode of the PMD controller is saved - in other words whether axes are enabled, the position loops are active, amplifier output is enabled, etc…

.PMD project files are not user-readable. They encode the project configuration information in a format that can be read (and written) by Pro-Motion, but is not intended to be read or edited by the user.

3.9.1   Project Configuration Restore

To call up a saved project select *File/Open Project*. You will be asked to specify a project file. Once you specify a file the content will be parsed by Pro-Motion and sent to the appropriate PMD controller(s).

Once saved configuration settings are loaded Pro-Motion will attempt to return each axis of the system to the operating mode it had when the project was saved. Along those lines, if appropriate, dialog boxes will appear asking whether the motor is ready to be energized. In the case of a Brushless DC motor energizing means the commutation is initialized and depending on the motor control mode setting the current loop and the position loop may be enabled. For DC Brush motors, similarly, the position loop and current loop may be enabled depending on the control systems in the project file. For step motors the drive current or holding current will be applied.

With communication connections and controller settings restored, after the project open operation the system should be restored to the same condition it had when the project was saved. At this point further development work on the project can occur and subsequent configuration changes stored, if desired, via the *File/Save Project* function. Alternatively a new project file can be created by saving the configuration via *File/Save Project As*.

Selecting File/Open Project to restore a saved configuration should be done with the connected PMD controller(s) in a reset condition or when in a stable non-moving state. Systems that have axes that may move when de-energized (such as vertical axes subject to the force of gravity) should only be restored from a reset condition. Failure to observe these guidelines may result in damage to the controlled mechanics.

3.10   Configuration Export to C-Motion

Pro-Motion provides the ability to export the Pro-Motion motion control setup as a C-Motion source code file, suitable for input to PMD's user application code building system. To learn more about how this export operation can be used to help with user application code building see the *readme.txt* file in the SDK you will be using for code development. For more information on C-Motion SDKs see <u>Section 6.1.4, "C-Motion SDKs."</u>

Here is some additional information that you may find helpful:

- To save the configuration to a C-Motion file use the menu function *File/Export Motion IC Settings as C-Motion*. A default file name is provided which can be changed. The specified C-Motion source code file will be created and displayed in Windows Notepad for convenience.

- For C-Motion exports there are generally more parameters saved than are actually used in a given application. If desired, the user can therefore review the saved configuration data and remove unused setting entries.

3.11   Pro-Motion Application Notes

In the next few sections we provide a few examples of Pro-Motion Application Notes illustrating how Pro-Motion can help characterize and optimize the operation of your motion control system.

3.11.1   Application Note — Determining the Optimum Trajectory Acceleration

Many different control parameter optimizations can be explored using the scope function's extensive list of traceable variables. In the sequence below we will provide an example of an optimization session that shows how to determine the profile acceleration setting. Our goal is to determine what maximum acceleration the motor and attached mechanisms can achieve without exceeding the safe current limit specification for the motor. In this example we will assume the motor is driven with a position loop, which is the normal positioning control mode for DC Brush and Brushless DC motors.

1 Start by opening up the Trajectory dialog box, which can be accessed from the Axis Control window. Specify Trapezoidal profile mode.
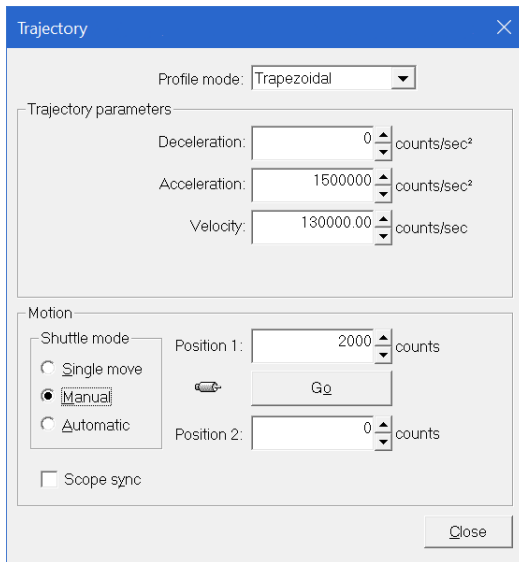
2 In the Shuttle mode box at the lower left select Manual.

3 Enter an acceleration value, a velocity value, and two destination positions. A deceleration of 0 commands the deceleration rate to match the acceleration rate. The two entered positions are the positions the motion shuttle will alternate between.

4 Next open up the scope window and enter the following settings:

Trace variable 1 - Active motor command

Trace variable 2 - Commanded velocity

Trace variable 3 - Position error

Trace Period: 10 cycle

Trace mode: One-time

Start trigger condition Activity Status register, In motion bit, State = 1

Stop trigger condition - Immediate

5 Review the trace length field located in the middle bottom of the screen and increase or decrease as desired. 250 to 500 is a typical number because it generally displays one oscilloscope screen-full without 'scrolling'. But most PMD products offer far more storage than this. Larger traces may be useful if data will be exported for analysis in a spreadsheet or other analysis tool. To export traced data to a file use the Pro-Motion *File/Export Trace* menu function.

6 Select the Start Trace button in the upper left of the trace window. The data graph display should not change. If it immediately begins to graph data wait till graph display completes and select Start Trace again. When in the proper state the graph area should go blank and although the Start Trace has been activated trace capture (or graphing) does not begin.

7 From the Trajectory dialog box select Go. You should see the trace data graphing area immediately begin to display captured data and continue till a full buffer has been captured. The reason this is the case is that by pressing Go the Magellan trajectory generator is activated and the Activity Status Register In Motion bit goes from 0 (false) to 1 (true).

To manage the data being displayed you can use the scope's **-** or **+** buttons to expand or reduce the horizontal scale. Alternatively if you would like to speed up the trace buffer update, see Section 3.8.1, "Trace Buffer Display Optimization," for suggestions.

Based on the resultant graph display you may also want to change the shuttle move distance so that the entire trapezoidal move is displayed.

The goal for setting an optimal trajectory acceleration is to increase it just until the PMD controller's programmed current limit (generally set equal to the maximum current specification on the motor data sheet) is exceeded. When this happens, even though the trajectory generator asks for more torque to achieve the requested acceleration, the torque output saturates resulting in the actual motor position rapidly falling behind the commanded position.

In this trial and error approach you will iterate by increasing or decreasing the setting being explored, re-run the profile and repeat until you find an optimum value. This process is made easier with the Manual shuttle feature of the Trajectory dialog box which uses two different programmed positions.

Each time you hit the "Go" button the commanded position switches from one to the other. Using the shuttle also has the benefit of creating alternating positive and negative direction moves, which is useful to confirm there are no unexpected differences in the movement based on the direction of motion.

The image below shows an example of what a trace may look like when optimizing the acceleration:
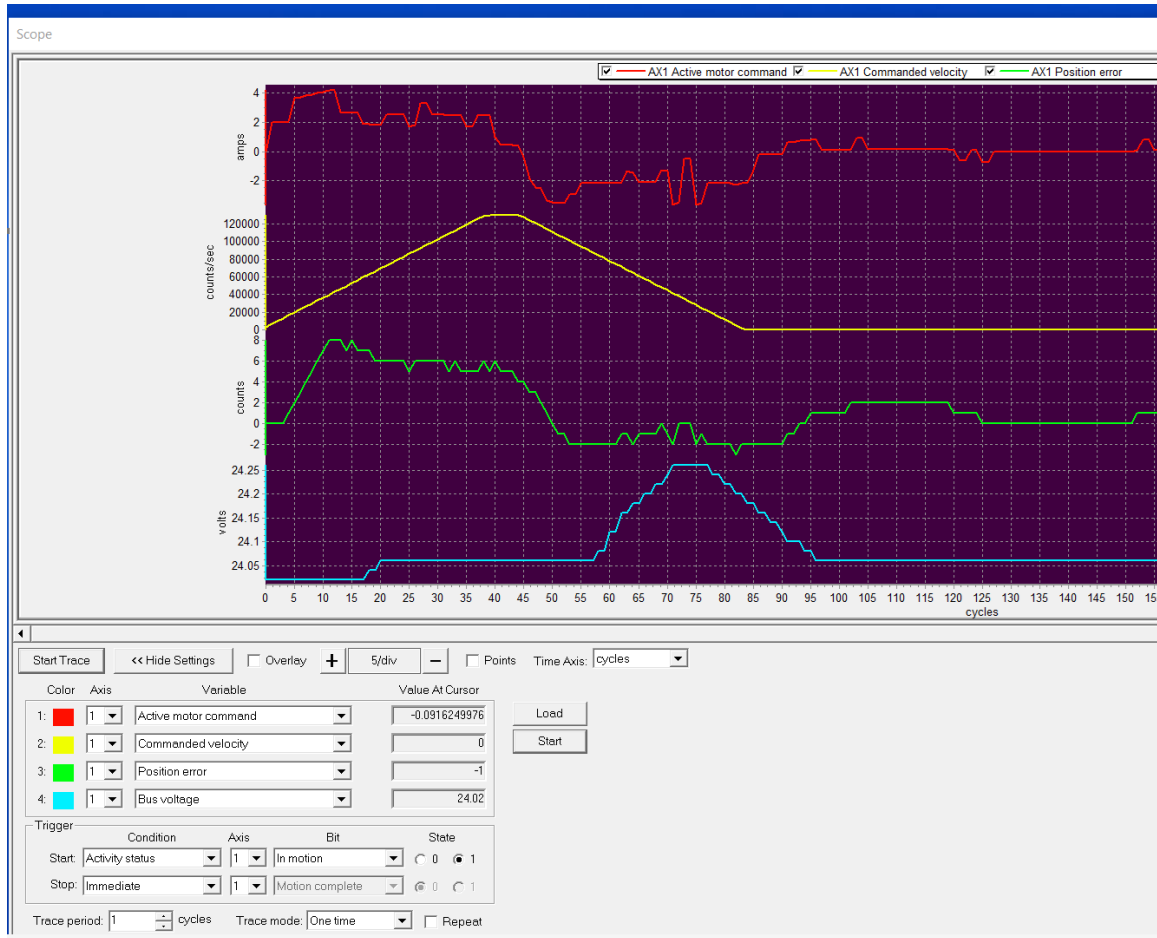
This screen capture shows data from a motor with a current drive limit of 4 amps, optimized to accelerate as fast as possible without exceeding the current drive limit. The resultant point-to-point trapezoidal move traverses 300 counts (~ 25 motor shaft degrees) in approximately 4 milliseconds. Note that despite this aggressive move the on-the-fly position error never exceeds 8 encoder counts, and the motor settles to within 2 counts error after 85 cycles, which corresponds to 4.25 mSec.

3.11.2 Application Note — Tuning the Position Loop

This application note provides an approach to developing PID position loop gain settings. Note that this section only applies to servo motors (DC Brush, Brushless DC) and motors using a closed loop stepper method.

Many users will first experience manually tuning their position loop while using the Axis Wizard. All servo motors (DC Brush or Brushless DC) which use the position loop require position loop gain settings. If you are tuning the position loop from the Axis Wizard make sure to enable the scope display by hitting the "Display Scope" button, after which you may skip to

If you are tuning your position loop from the Axis Window, either because you didn't set up the motor configuration using the Axis Wizard, or because you want to re-tune gain parameters for your motor, continue reading from the next section to set up the scope window for a tuning session.

3.11.2.1 Scope Window Tuning Setup

1 To set up a tuning session the motor should be energized. Open the position loop dialog box by clicking the Position Loop box in the Axis Control window. The diagram below shows what this dialog box looks like:

2 Next, click the Tuning button on the right side of the top icon bar of Pro-Motion. After doing so the trace scope will be displayed along with a Step Response dialog box. You may want to adjust the location and size of the Scope window to fit your monitor size.

3 In the Scope window select *Active Motor Command* for the first variable to trace, select *Commanded Position* as the second variable, and select *Actual Position* as the third variable.

3.11.2.2 Position Loop Tuning

With the scope properly set up for a tuning session (or if you are tuning from the Axis Wizard where the scope is already set up for tuning) the following instructions apply.

1 Specify a step move distance. A typical move distance for motors with an encoder is 50 or 100 counts. For motors using Halls to track the position a typical move distance is 5 to 10 counts.

2 Set Ki (Integration gain), Integration limit, Kaff (Acceleration feedforward gain), and Kvff (Velocity feedforward gain) to zero. Set Kp (Proportional gain) to a small value, typically 10 to 50 for motors with encoders and 100 to 500 for motors with Halls only. Set the Kd (Derivative gain) to a value 5 times greater than the Kp value you entered. Set the derivative time to 5 and set KOut to 100%.

3 Click the right arrow key. This should result in the motor quickly (in a single instantaneous step) moving by the programmed amount, and shortly thereafter the trace scope window should begin to display data. If this doesn't happen review the instructions above and try again.

4 If the motor oscillates in an uncontrolled manner set Kp to a lower value and click the right arrow key again. If at any point you are concerned that the motor or power supply may be over-stressed you can hit the "stop" icon in the tool bar to immediately disable the servo loop. If the motor produces a high-pitched chatter or whine you can lower the KD derivative gain, or alternatively you can increase the

derivative time and proportionally lower the Kd value. For example if the derivative time was 5 with a Kd of 1,000, changing the derivative time to 10 means the Kd should change to 500.

Once the position loop is at least somewhat stable, you will begin a repeating sequence consisting of:

- View the scope display to determine whether the response is underdamped, critically damped, or overdamped (see below for instructions on how to do this)

- Adjust the gain settings accordingly

- Make another step move

The next three sections will show you how to characterize the step move response which will in turn indicate how gain settings should be adjusted.

> During servo tuning a certain amount of oscillation is not unusual, but if at any point you become concerned that the motor or power supply may be over-stressed hit the "stop" icon in the tool bar to immediately disable the servo loop.

3.11.2.3   Underdamped Response

The screen capture below shows a typical underdamped response.
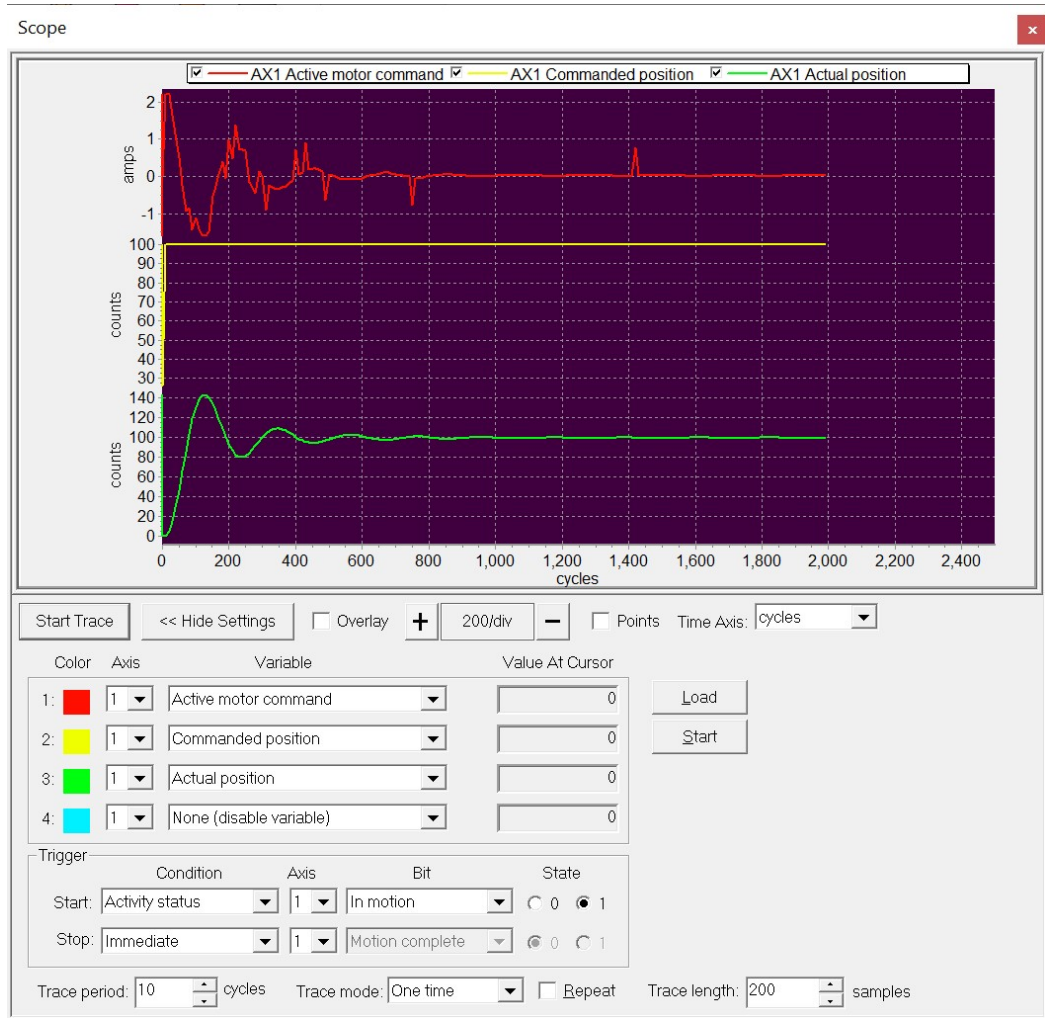
The yellow trace shows that the commanded position instantly changes from a position of 0 to a position of 100 (in your setup this jump will equal whatever you entered into the step response dialog box). The green trace shows the resultant actual motor location.

From this trace, although the actual position does eventually settle to a position value of 100, it overshoots significantly and oscillates several times. Whenever the actual axis position overshoots the commanded position the system is considered underdamped, and to correct for this you should increase the derivative gain (Kd). The goal of the Kd setting is to determine a value that results in a critically damped response, as shown in the next section.
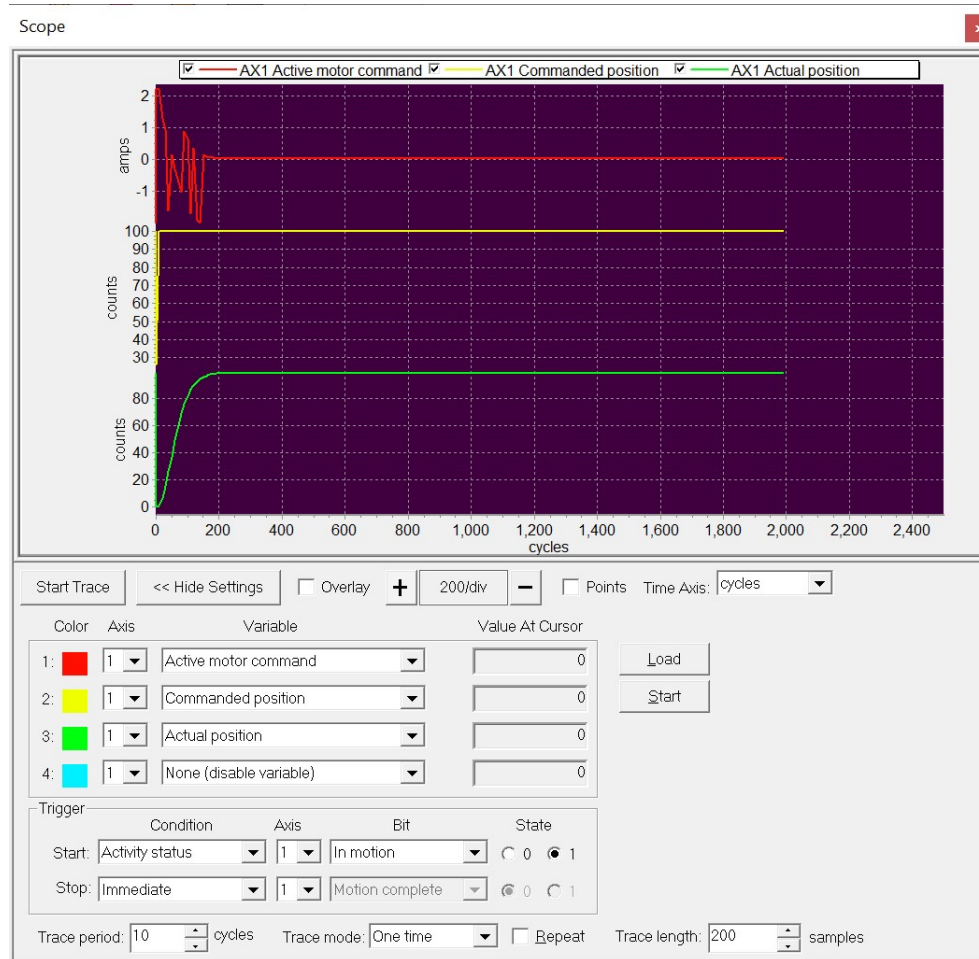
3.11.2.4 Critically Damped Response

The screen capture below shows a critically damped response.

In this trace the actual position does not overshoot that commanded position, and it settles quickly for this particular motor in 10 mSec to the commanded position of 100.

It's worth noting that many Kd settings could result in a response that appears critically damped. The distinction between critically damped and overdamped is somewhat subjective when using this manual trial and error process. In general you should try to find the smallest setting of Kd which still provides a critically damped response.

3.11.2.5   Overdamped Response

The screen capture below shows an overdamped response.
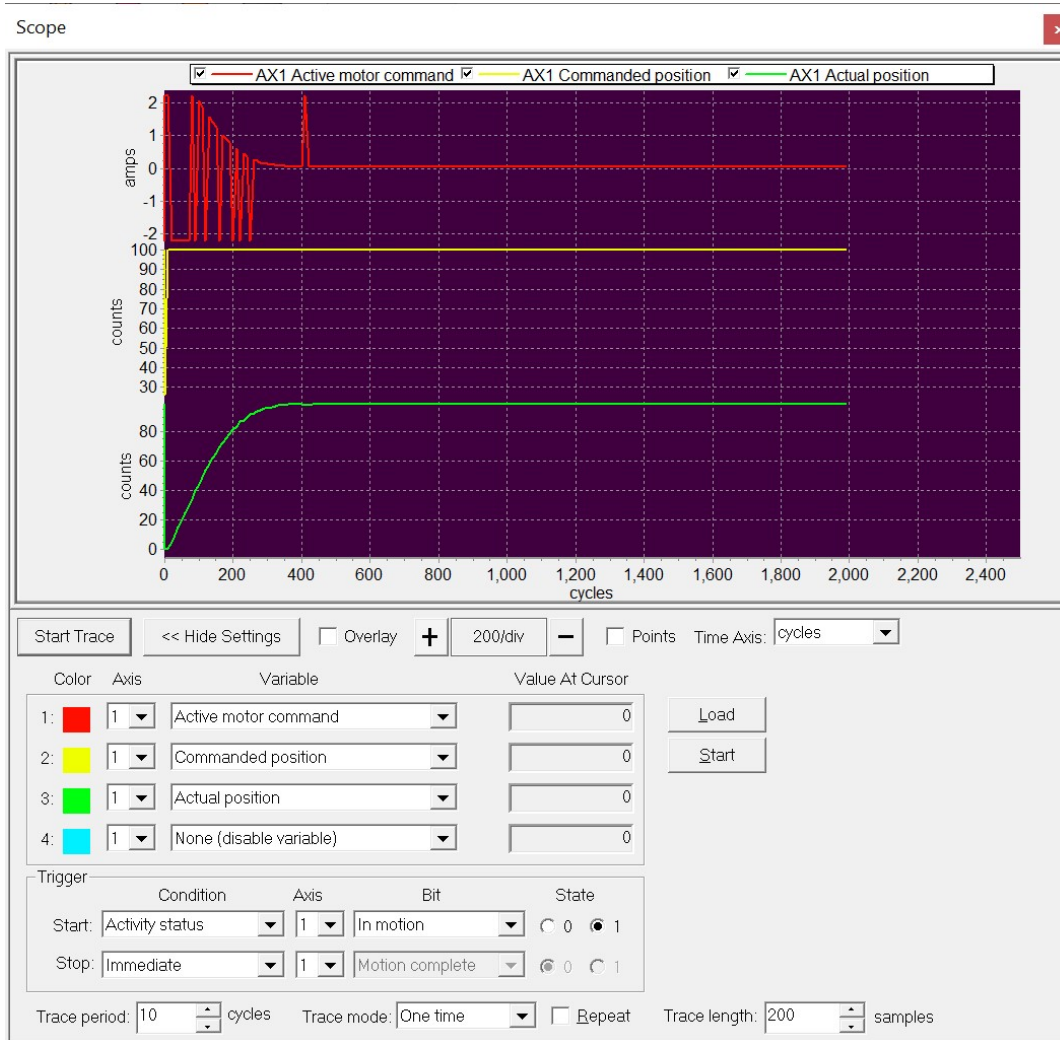
This can be seen by comparing with the critically damped response in the previous section. An overdamped system will be stable but needlessly slow in its response to changes in the position command.

If the response is overdamped you should reduce Kd, trying to find the smallest value of Kd that still gives a critically damped response.

> For purposes of illustration in the above three example graphs the Kp setting was 100 and the Kd settings were 2,000, 6,000, and 25,000 for the underdamped, critically damped, and overdamped responses respectively.

3.11.2.6   Getting To Final Gain Settings

Your overall position loop tuning goal is to set the Kp to as high a value as possible while still finding associated Kd values that can create a critically damped response. Higher Kp values will result in more accurate tracking and faster responses to command position changes.

As you try higher and higher Kp values, at some value of Kp you will find that there is no value of Kd that can create a stable and critically damped response. At this point you should reduce the Kp setting by 35-50% and determine the associated critically damped Kd setting. Backing down from the 'borderline' Kp value is important for improving stability, accommodating small differences in controller, motor, and attached hardware behavior, and for handling changes that may occur to the system over time.

3.11.2.7   Setting Derivative Time

The Derivative Time setting is the period at which the PID loop derivative contribution is calculated in units of sample time cycles. Increasing the Derivative Time can be useful for reducing axis 'chatter' by effectively introducing a low pass filter on the PID loop's control response. This in turn allows the effective Kd damping contribution to be increased.

The impact of the Kd on the PID loop command output is a direct multiple of the derivative time. For example if the derivative time is set to 10 and the Kd value is 100 this would be an equivalent impact as a derivative time of 5 and a Kd of 200. Generally speaking larger motors with larger loads will use higher Derivative Time settings and vice versa for smaller motors with smaller loads.

3.11.2.8   Setting Ki and Ilimit

Once Kp and Kd have been set, you can enter a value of Ki (Integration gain) to improve tracking accuracy. Typically, Ki settings are 1/10 to 1/2 of the Kp value. Smaller motors typically use relatively large ratios, and larger motors with larger loads typically use smaller Ki ratios. Higher Ki settings will increase tracking accuracy during and after the motor move is complete but can also reduce system stability. So you should set Ki to the smallest value that can achieve your goals for final or dynamic tracking while not distorting the critically damped response the Kp and Kd settings created.

The Integration limit can be set to a high value, perhaps 10,000,000. The Magellan IC PID engine has built in anti-windup logic so it is very unlikely this integration limit will ever be reached.

3.11.2.9   Frequency Based Tools & Analysis

The above process represents a popular and relatively straightforward approach toward tuning the PID loop. There are more sophisticated approaches however both for determining PID parameters and for verifying the behavior of a given position loop controller. Pro-Motion provides frequency-based tools known as Bode plots to support such methods.

Bode plots can be used in different ways and can help characterize your mechanical system as well as determine important characteristics of the operating control loop such as the bandwidth and phase margin. The Pro-Motion Bode plot tool can be accessed from the *View/BodePlots* Pro-Motion menu selection.

One output of a Bode-based system characterization may be the identification of natural resonances in the mechanical system. While a properly tuned PID can help reduce the impact of this, the Magellan IC also provides two general purpose biquad filters in the position loop. Biquad filters can be used to create low pass and notch filters. For more information on biquads as well as the Magellan Position PID loop refer to the *Magellan Motion Control IC User Guide.*

3.11.3   Application Note — Determining Acceleration Feedforward Gain

The following optimization session using the Scope window shows how to determine the acceleration feedforward gain.

The graph below shows a complete point-to-point profile move using the critically damped system settings from The yellow line represents the commanded trajectory velocity, and the green line represents the position error (the difference between the commanded and the actual position). Notice that despite the fact that the axis is accelerating and decelerating quite aggressively (the entire move

duration is less than 200 mSec) the motion is stable throughout and the tracking accuracy is quite good even during the motion, varying from +8 counts maximum position error to -5 counts.
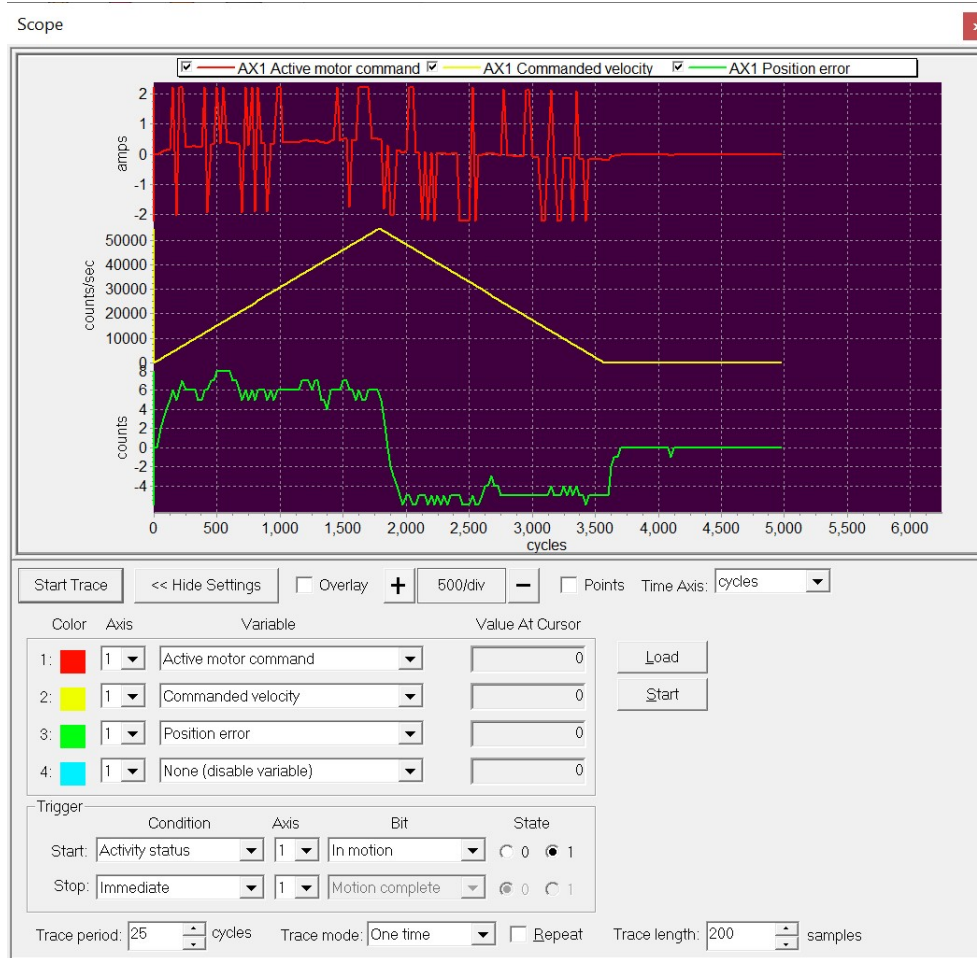
Nevertheless Acceleration feedforward (Aff for short) can reduce the dynamic tracking error even further. Acceleration feedforward works by adding a torque command proportional to commanded acceleration. PMD's Magellan IC also supports velocity feedforward, which similarly adds to the motor command proportional to the commanded velocity. Velocity feedforward is used less often than acceleration feedforward, typically to compensate for fluid friction such as lubricated bearings or in fluid pumping applications.

The reason Aff improves tracking accuracy is that from the position error graph above we can see that the form of the position error mimics the form of the trajectory's commanded acceleration. This can be seen in Figure 3-1. Therefore by 'pre-injecting' this feedforward value into the position PID loop the servo is required to do less work, resulting in better tracking accuracy.

**Figure 3-1:
Velocity and
Acceleration
Versus Time for
Point-To-Point
Trapezoidal
Trajectory**
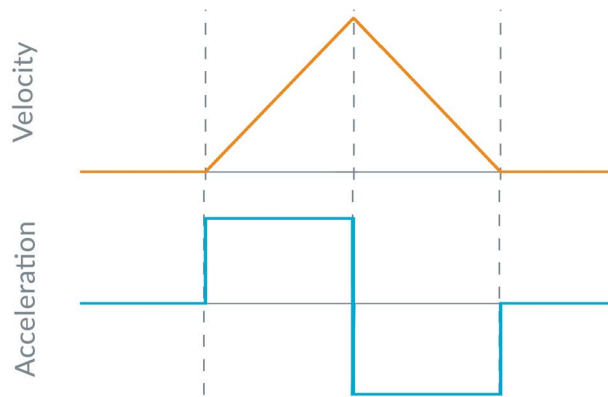
This general session for optimizing Acceleration (or Velocity) feedforward uses the same setup as described in Section 3.11.2, "Application Note — Tuning the Position Loop."

As for all trial and error optimizations with the Pro-Motion scope function you will iterate by increasing or decreasing the setting being explored, re-run the profile and repeat until you find an optimum value. This is most easily accomplished using the Manual shuttle mode of the Trajectory dialog box. See Section 3.11.2, "Application Note — Tuning the Position Loop," for details.

In the case of determining the best Aff value, you will find that when you are below the best Aff value further increasing the gain decreases the average position error, and when you are above the best value further increasing the Aff gain increases the position error. You are looking for that 'best' Aff value where any increase or a decrease in gain value increases the position error.

After adding an optimized Aff gain to the PID loop the results are shown below. Although the tracking is not perfect, the magnitude of the position error is halved during the motion.
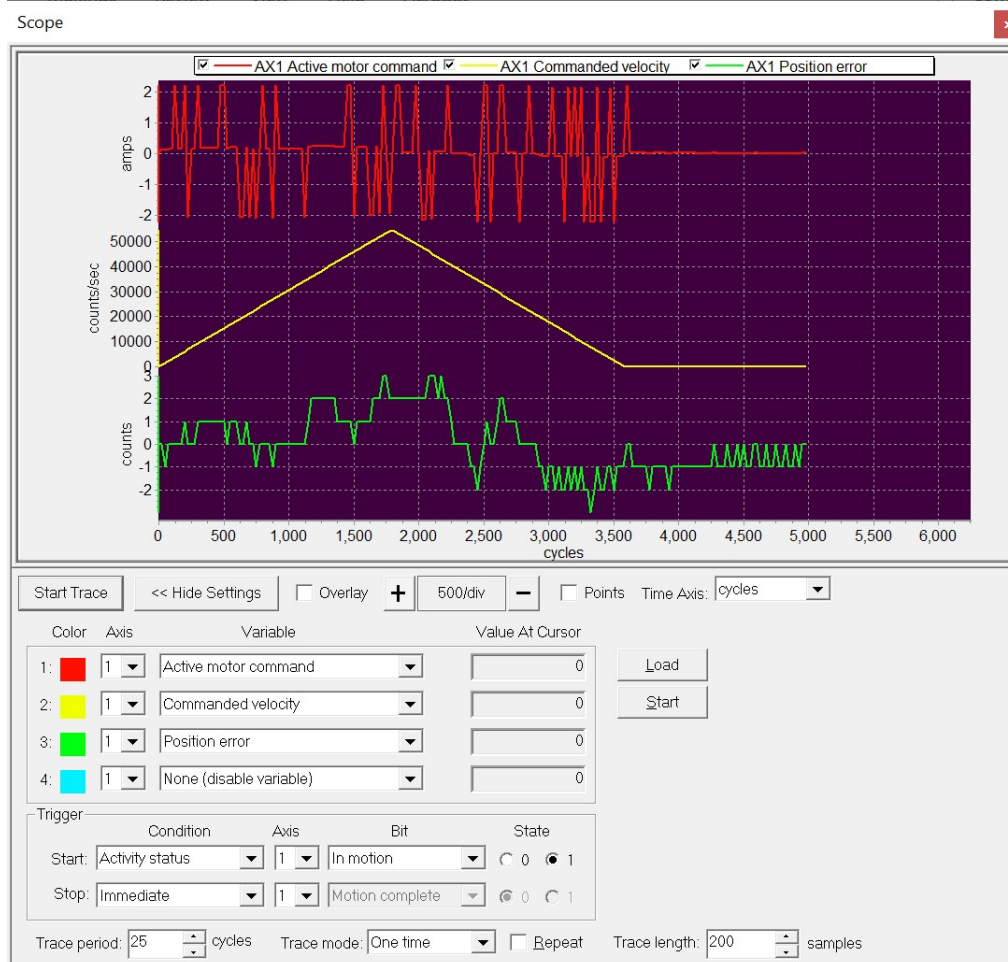
One caution with acceleration feedforward is that it is load specific. If you tune the Aff value by increasing and decreasing till the net position error during motion is minimized, you will find that if the load on the motor changes you will need to re-tune the Aff value to achieve a similar result. This means systems that typically carry a variable load may not be good candidates for using an acceleration feedforward gain.

3.11.4 Application Note – Monitoring and Enhancing DC Bus Voltage Stability

An important aspect of designing a robust motion control system is ensuring stability of the power supply voltage. This can be challenging because motors, as they accelerate and decelerate, may demand rapid increases in current from the power supply, or may actually send current back to the power supply.

In this application note we will walk through how to characterize the stability of the voltage supply under various motor operating conditions, and then suggest techniques for improving voltage stability.

There are three HV supply voltage conditions to consider for a given combination of power supply and motor operation:

1 Motor and load trajectories that result in stable operating voltage of the DC supply.

2 Motor and load trajectories that result in undesirable voltage drops because the DC supply's current output capacity is exceeded.

3   Motor and load trajectories that result in undesirable voltage increases because the DC supply cannot absorb any, or enough, current from the controller.

3.11.4.1   Stable DC Supply Voltage Operation

In this section we will use the trace capability of PMD controllers and Pro-Motion to show what stable operation of the DC supply looks like while the motor is moving. We will perform a simple point to point trapezoidal profile move while monitoring the voltage level of the DC supply.

All PMD controllers that provide amplification have the capability to measure the DC bus voltage directly and to trace this value at high speed along with other motion parameters. The resulting scope trace is shown below.
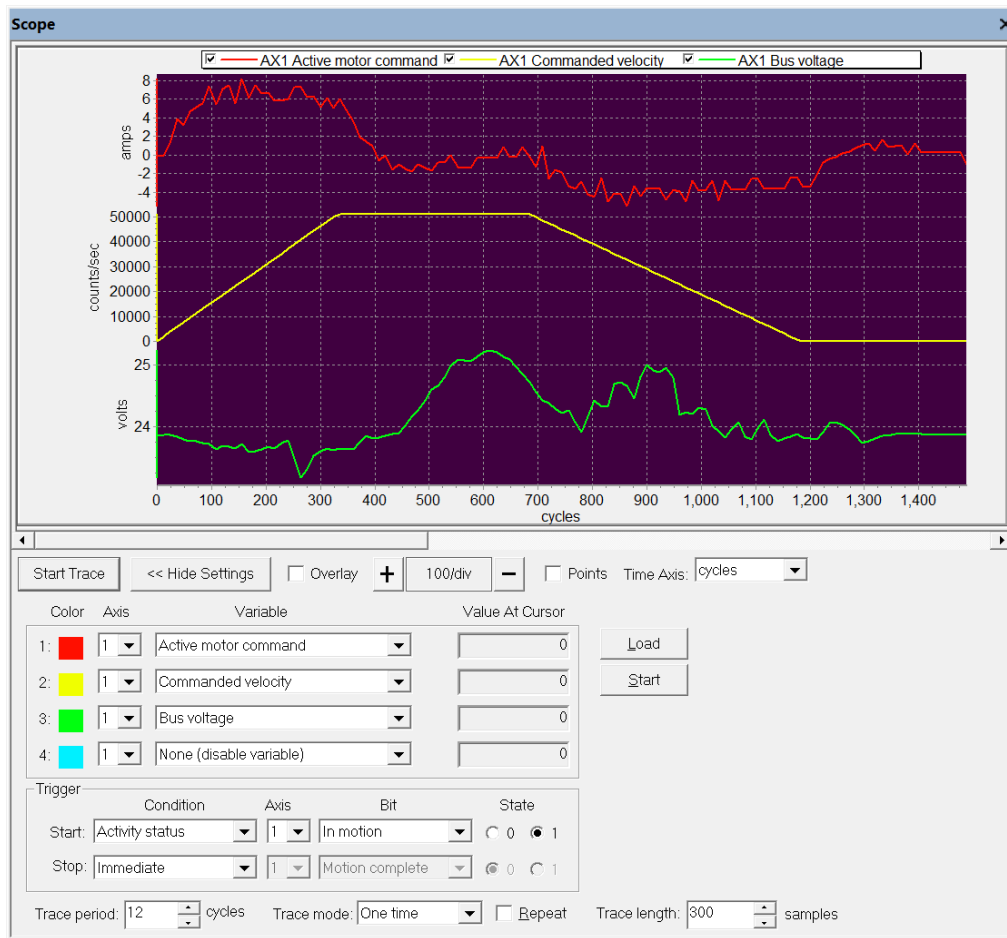
In this stable voltage scenario the main feature of interest is that as the motor accelerates and decelerates, although there is a variation in the 24V supply voltage, its magnitude is small - in the above trace between 23.2 and 25.3 volts. This level of variation is acceptable in nearly all motion systems, resulting in little stress on the power supply, and in no meaningful impact on the controller's ability to maintain accurate position control through the move.

3.11.4.2   Undervoltage Condition During Motion

In the Pro-Motion scope trace below the same motor setup and load used with the scope trace above is driven at a higher acceleration, resulting in a significant drop of the supply voltage from 24V to 18V. The trajectory's coasting and deceleration phase allows the power supply to return to a stable voltage supply situation.
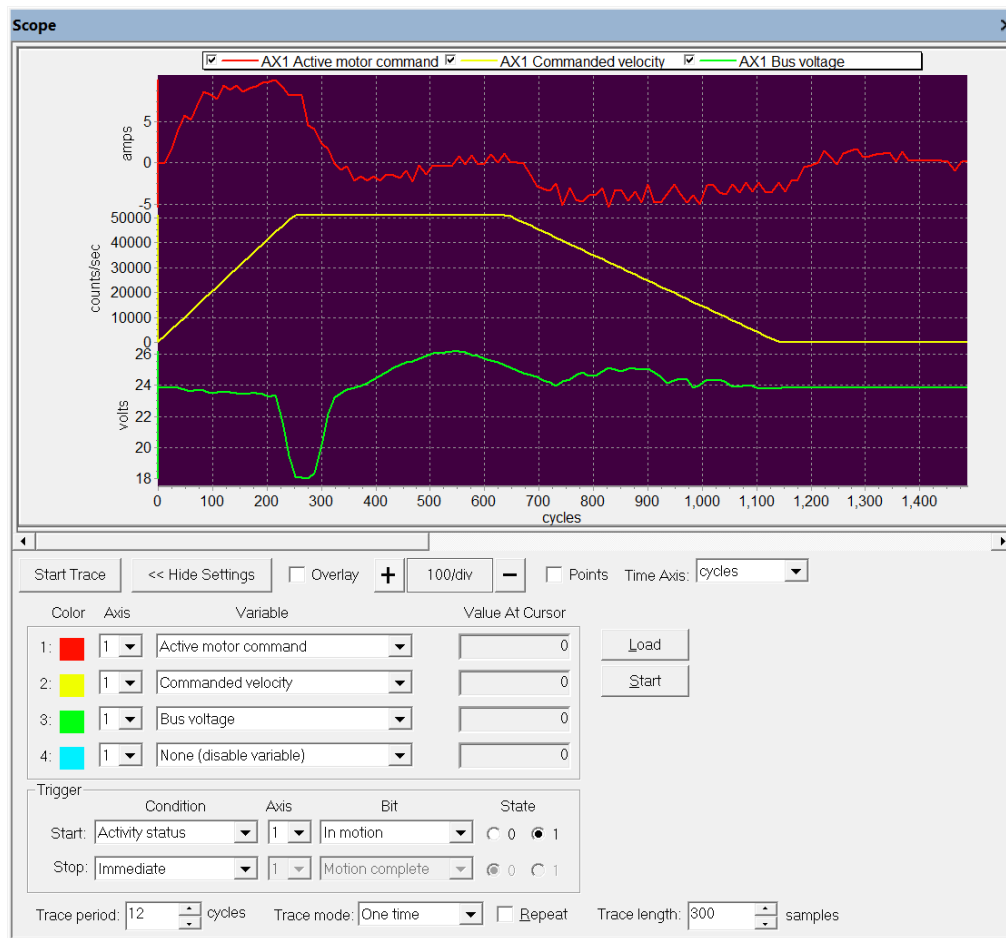
Why does the voltage decrease during trajectory acceleration? The reason that the supply voltage drops is that during acceleration the motion controller rapidly commands more current to the motor, which in turns means more current is 'requested' from the DC supply. The power supply cannot provide this dramatic increase in current and the result is a drop in voltage, an effect similar to an electrical grid brownout.

The current commanded by the motor controller is directly proportional to the amount of torque needed to accelerate the motor and load, with the torque being proportional to acceleration times the moment of rotational inrertia. For example if the acceleration doubles and the load stays the same the amount of torque (and therefore required current) doubles. The significance of this is that to establish the supply voltage stability for a given mechanism the worst case current demand should be investigated. For a single axis machine this would be the highest acceleration move with the heaviest load, and for multi-axis machines this would be the same, but including all axes that move simultaneously.

Why is a drop in voltage a problem? There are a number of reasons why a voltage drop may be a problem. The first is that the power supply may become overstressed. Depending on its design, a request for current that exceeds its output capacity, may overheat or damage the power supply.

Another problem is that servo stability may be impacted. In extreme cases of voltage drop the current control loop or the position control loop may become unstable. Finally, since motion controllers generally derive their internal logic power from the main supply voltage, if that supply voltage drops too far the controller may cease functioning. Modern motion controllers (including PMD's products) have a feature called undervoltage protection which automatically

disables the motor drive and shuts down the trajectory generator. When an undervoltage event occurs, although the move is aborted, the more dangerous situation where the motion controller completely shuts down is avoided.

What are the remedies to voltage drop during motor acceleration? The simplest solution is to reduce the acceleration rate. This will directly reduce the amount of current the power supply needs to deliver. See Section 3.11.1, "Application Note — Determining the Optimum Trajectory Acceleration," for an application note on this exact subject.

If reducing the acceleration is not an option the next two options to consider are switching to a power supply with greater current output or adding capacitance to the DC supply line. Especially to provide short bursts of additional current during acceleration, adding capacitance is often the preferred option.

One more option for reducing supply voltage sag during motor acceleration is to switch to a motor that can output more torque for a given current. Note that this might also be accompanied by the need for an increase in the operating supply voltage.

3.11.4.3 Overvoltage Condition During Motion

The scope trace below shows a setup similar to the one above except with a higher deceleration trajectory resulting in a significant increase in DC supply voltage. Overvoltage conditions are potentially dangerous because they may damage the power supply, motion controller, or motor, or may lead to instability of the motion controller's servo control function.
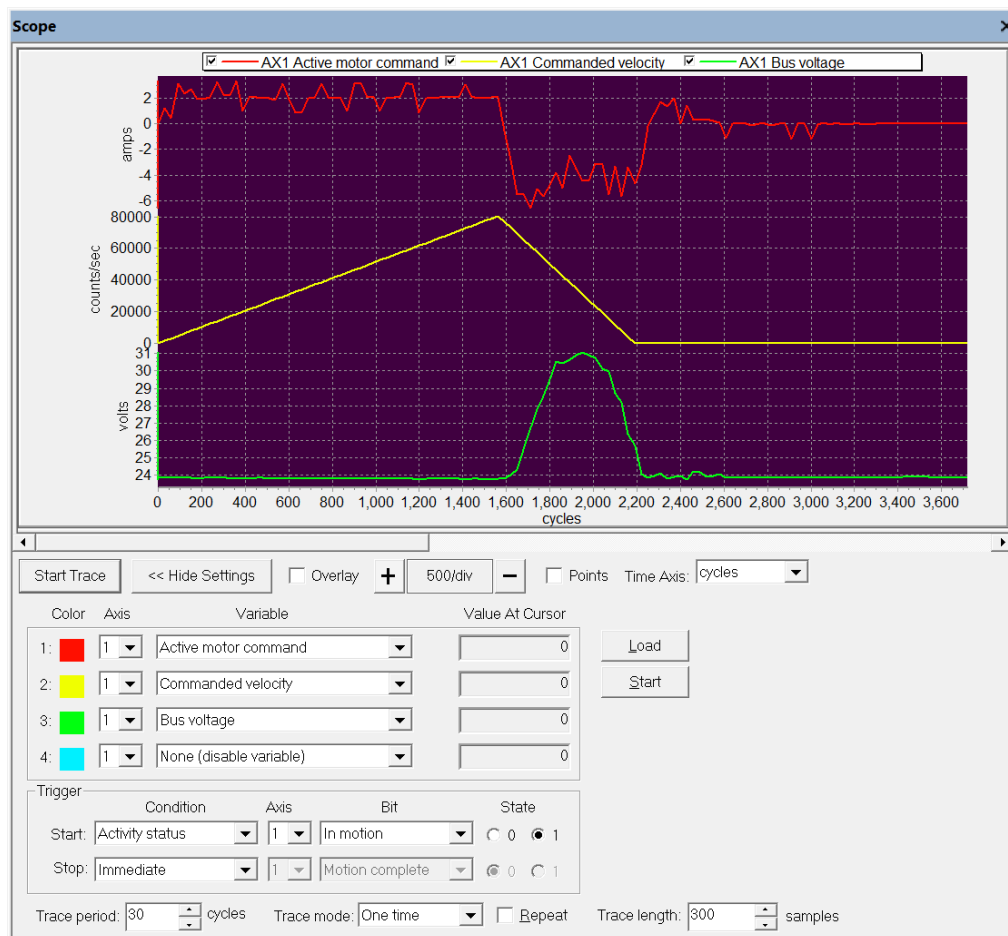
Why does the voltage increase during trajectory deceleration? The reason that the voltage increases during deceleration is that the motor acts like a generator. During deceleration the mechanical energy stored in the rotation of the motor and attached load is converted to electrical energy which must be absorbed somewhere. This negative current output has

the effect of increasing the supply voltage because most regulated power supplies have very little capacity to absorb energy from the device they are powering.

Does trajectory deceleration always generate negative current flow? No. The motor becoming a net generator occurs at higher deceleration rates for motors & loads with larger rotational inertia. An additional factor that affects this is the amount of friction in the system. Predicting when a mechanism will become a net generator is complicated so most engineers exercise the system and directly observe the supply voltage.

What are the remedies to voltage rise during motor acceleration? If an undesirable voltage rise is seen in the deceleration phase of the trajectory the simplest solution is to reduce the deceleration rate. This will reduce or eliminate the power supply needing to absorb current. Another option is to add capacitance to the HV DC supply. This is especially effective to absorb short bursts of current.

A third option is called shunt regulation. Shunt regulation functions by continually comparing the DC bus voltage to a user programmed threshold (usually set to a few volts above the supply voltage) and 'shunting' or diverting power directly from HV to GND when the threshold is exceeded. A resistor and diode in the shunt circuit path controls the rate of current flow and ensures that current flow is always in the correct direction. The power shunted through the resistor causes the resistor to heat up, therefore care should be taken to ensure that the resistor and any heat sinking elements it is mounted on can dissipate the worst case heat load.

See Section 4.7, "Shunt Regulation," for more information on using this PMD controller's shunt function.

In the trace below the same exact motion profile as above is executed with a shunt function in place with the voltage threshold set for 25.0V. As can be seen shunt does an excellent job of regulating the voltage, lowering the maximum voltage rise from 31V to 25.3V.
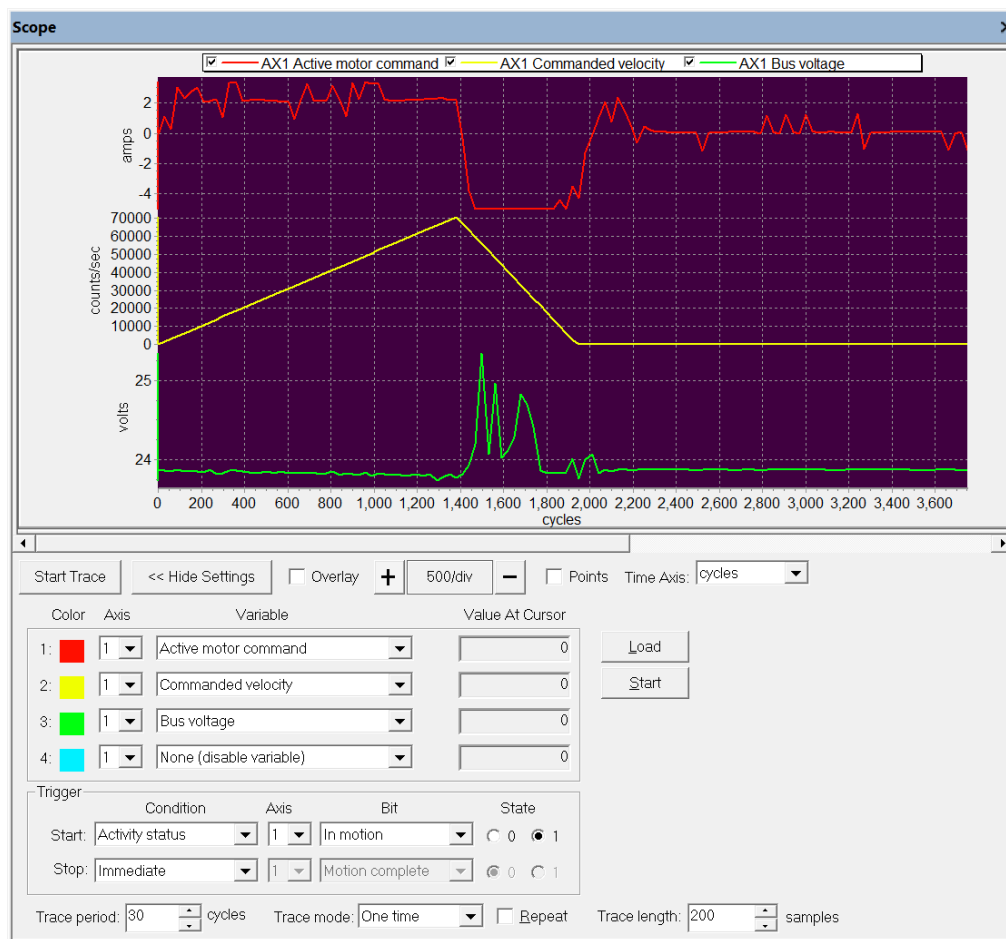
3.12 Troubleshooting Suggestions

If the first-time verification sequence detailed in <u>Section 2.4, "Step #4 — First Time System Verification,"</u> was successful problems while using Pro-Motion are not common. Nevertheless below is a list of suggested corrections for issues users may encounter while exercising their motion hardware.

The Motor Stops Responding. If a motor previously moving and responding to your commands stops responding the most likely reason is that the operating mode is not set correctly. Although most such 'transitions' from normal operation to an error/protection state result in a dialog box appearing, to check the operating mode select the Operating Mode box in the Axis Control window and adjust the settings if needed. See <u>Section 3.4, "Axis Control Window,"</u> for more on the Operating Mode dialog box.

The Step Motor Stops Responding. In addition to checking the operating mode as detailed above, step motors controlled via microstepping mode may stop moving because the motor command has been set to zero. This may occur during safety related actions or when certain command functions are sent. To check and restore the motor command setting select the Motor Control box in the Axis Control window.

Events Aren't Being Reported. As the Magellan Motion Control IC executes its control settings some occurrences representing a change in the control system state may occur. These changes are called events and include items such as motion error, over temperature, breakpoints, and more. To check the Pro-Motion settings for which events will result in a dialog box popping up click the Event Manager box in the lower right of the Axis Control window. For detailed information on Magellan events refer to the *Magellan Motion Control User Guide*.

An Overvoltage Event Occurred. If during a motion profile an overvoltage event is indicated the most likely reason is that during deceleration of the motor the inertia of the motor and load resulted in net generation of energy. Depending on the design of the power supply you are using this can result in the HV voltage rising and eventually exceeding the overvoltage limit. Remedies are to reduce the profile's rate of deceleration or add more capacitance to the HV supply.

An Undervoltage Event Occurred. If during a motion profile an undervoltage event is indicated the most likely reason is that during acceleration the power supply was not able to deliver the needed amount of current, or was not able to deliver an increase in current fast enough. Undervoltage (and overvoltage) events may also occur if the current loop or position loop is unstable. Remedies are to increase the current output limit/rating of the power supply, add more capacitance to the HV supply, or reduce the profile's rate of acceleration or re-tune the servo loops.

An Overcurrent Event Occurred. If during a motion profile an overcurrent event is indicated the most likely reason is that during a rapid trajectory deceleration the inertia of the motor and load resulted in the HV voltage rising rapidly, which, in combination with a motor with very low coil resistance can sometimes result in excess bus current. Remedies are to reduce the profile's rate of deceleration, use the shunt feature of the PMD controller (if it has one), or add more capacitance to the HV supply.

A Motion Error Event Occurred. If during a motion profile a motion error event is indicated there are a few potential reasons. Motion error, also called position error, means the difference between the commanded position and the actual motor position has exceeded a user-specified threshold. Potential reasons are that the axis motion was blocked, that the position loop is not well tuned, that the commanded profile velocity exceeds the achievable motor velocity, or that an aggressive profile has temporarily resulted in a motion error during the move. Remedies depend on the cause, and are generally straightforward once the cause is determined. To change the position error threshold use the Tracking box in the Axis Control window.

Communication Errors Are Occurring. Communication errors between Pro-Motion and the controller are rare, but depending on the connection type and motion setup may occur. The 3-pin Programming Port for example, used with some Juno IC DKs and N-Series ION DKs, is susceptible to noise if high current moves are commanded because it uses a UART signal scheme which is not a differential signal. For whatever communication link is used, you should

try to determine if there are specific operating conditions that result in communication errors. If signal noise may be at fault consider switching to a link that uses a more robust signaling scheme such as RS232, CAN, or Ethernet.

Depending on the type of amplifier used some of the events described above may not be reportable by the control system. For example if Atlas amplifiers are used Overvoltage and Undervoltage event reporting will occur correctly, but if user-designed or non-PMD amplifiers are used these events will not be detected or reported by Pro-Motion.

This page intentionally left blank.

4. DK58113 Board Operation

In This Chapter

▶ DK58113 Block Diagram
▶ Communication Ports
▶ On-Board Switching Motor Amplifier
▶ Drive Protection and Control Signals
▶ DC Bus
▶ Operating With External Amplifiers
▶ Shunt Regulation
▶ Motion Peripheral Signals
▶ Enable and FaultOut Signals
▶ Multi-board Synchronization
▶ MC58113 IC Reset

4.1 DK58113 Block Diagram

The DK58113 Developer Kit board provides a complete functioning MC58113-series IC exerciser and development system. It directly interfaces to a host computer using serial, CANBus, or SPI communication, and to all power and feedback signals required to drive a DC Brush, Brushless DC, or step motor.

The DK58113 incorporates several major subsystems including the MC58113-series IC itself, communications interface circuitry, a high performance MOSFET-based motor amplifier, DC Bus conditioning circuitry, and various other functions.

The following sections describe the operation of the DK58113 board. For a complete description of the MC58113 IC, see the *Magellan Motion Control IC User Guide*, *MC58113 Electrical Specifications*, and the *C-Motion Magellan Programming Reference*.

Figure 4-1:
DK58113 Block
Diagram
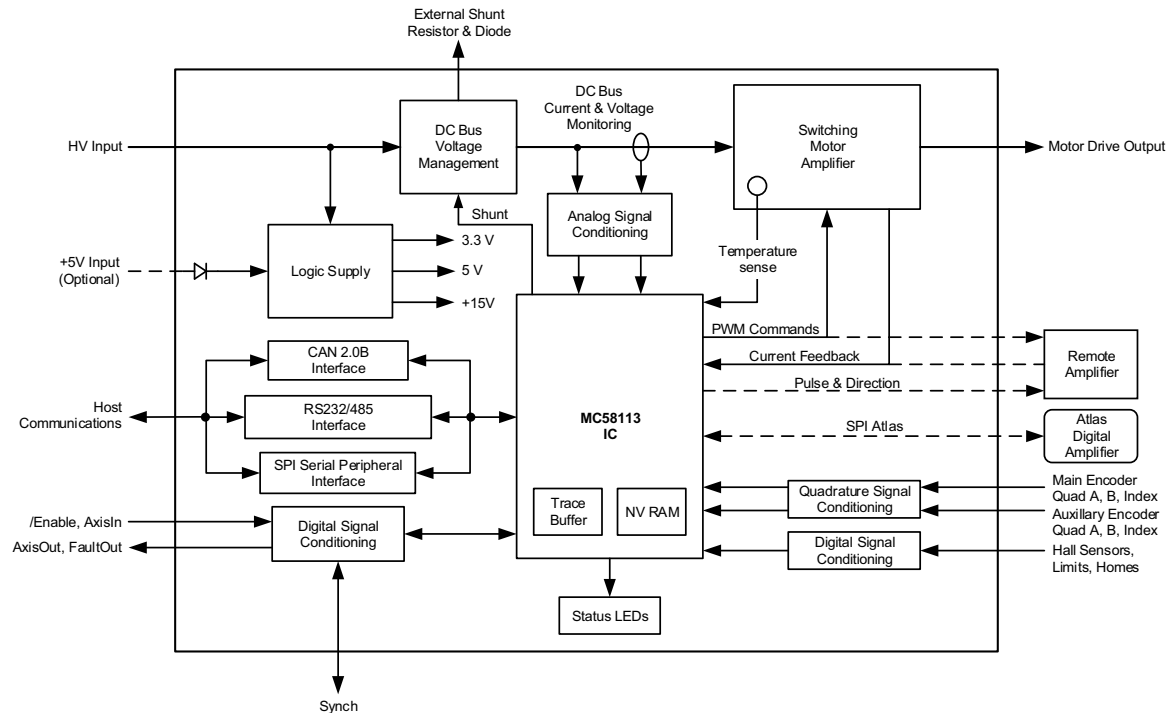
4.2   Communication Ports

4.2.1   RS232/485

The DK58113 board supports both RS232 and RS485 as well as both point-to-point and multi-drop networking. For RS485 the DK58113 board supports 4-wire full duplex operation. Jumper JP4 is used to select between RS232 and RS485 operation. For RS232 JP4 is connected at 1-2 (which is the factory default setting), and for RS485 operation JP4 is connected at 2-3.

Both RS232 and RS485 protocols can be operated at the communication settings shown in the following table:

Parameter	Range	Default
Baud rate	1,200 to 460,800	57,600
Parity	None, even, odd	None
# data bits	5, 6, 7, 8	8
# stop bits	1, 2	1

These communication parameter settings can be changed, see Section 5.1, "Serial Communications" for details.

All DK58113 board communication functions are controlled by the MC58113 IC. For complete information on the serial port function of MC58113 family ICs see the *Magellan Motion Control IC User Guide.*

For information on how to set up and communicate to one or more DK58113 boards via serial using Pro-Motion see Section 5.1, "Serial Communications."

> The DK58113 board does not have built-in termination for RS-485. If a network application requires termination at the serial connector, resistors should be added in the network wiring.

4.2.1.1 Connections & Associated Signals

All serial communication signals are carried on Serial Connector J5. The table below shows the signals associated with RS232 & RS485 communications:

Signal Name	Pin #	Description
	J5 – Serial Connector	
SrlXmt	2	RS232 transmit output
SrlRcv	3	RS232 receive input
GND	5	Ground
RS485Rcv+	6	Positive (non-inverting) RS485 receive input
RS485Rcv-	7	Negative (inverting) RS485 receive input
RS485Xmt+	8	Positive (non-inverting) RS485 transmit output
RS485Xmt-	9	Negative (inverting) RS485 transmit output

4.2.2 Serial Peripheral Interface

The DK58113 board supports an SPI (Serial Peripheral Interface) connection via its J11 connector. The user may use this connector to communicate with the DK58113 board from their own custom hardware or via a Windows-based PC.

The DK58113 board SPI port operates as an SPI slave. The host SPI interface consists of five signals; HostSPIEnable, HostSPIXmt, HostSPIRcv, HostSPIClock, and HostSPIStatus. Four of these are standard SPI bus signals and HostSPIStatus is a special signal used to coordinate command processing between a host and the MC58113 IC.

All DK58113 board communication functions are controlled by the MC58113 IC. For complete information on the SPI (Serial Peripheral Interface) port function of the MC58113 family ICs see the *Magellan Motion Control IC User Guide*.

There are no user-configurable parameters for the SPI interface. For information on how to communicate to a DK58113 board via SPI using Pro-Motion see

4.2.2.1 Connections & Associated Signals

All SPI communication signals are carried on Host SPI Connector J11. The table below shows the signals associated with SPI communications:

Signal Name	Pin #	Description
	J11 – Host SPI Connector	
HostSPIXmt	1	Host SPI bus synchronous transmit output
HostSPIRcv	2	Host SPI bus synchronous receive input
HostSPIClock	3	Host SPI bus synchronous clock input
HostSPIEnable	4	Host SPI bus active low enable signal input
HostSPIStatus	6	This signal indicates when an SPI response is available.
GND	7	Ground

4.2.3 CANbus

The DK58113 board provides a general purpose CAN port compatible with the CANbus 2.0 standard via its J6 and J7 connectors. The DK58113 board will coexist (but not communicate) with CANOpen nodes on that network. The DK58113 board uses CAN to receive commands, send responses, and (optionally) send asynchronous event notifications.

The CAN interface may be operated at various communication rates from 10,000 to 1,000,000 bps (bits per second) as shown in the following table, which also shows address ranges and defaults for the three supported address types:

Parameter	Range	Default
Bit rate	10,000 to 1,000,000 bps	20,000 bps
Send address	0x580 – 05FF	0x580
Receive address	0x600 – 0x67F	0x600
Event address	0x180 – 0x1FF	0x180

All DK58113 board CAN communication functions are controlled by the MC58113 IC. For complete information on CAN communication functions of the MC58113 family ICs see the *Magellan Motion Control IC User Guide.*

For information on how to communicate to one or more DK58113 boards via CAN using Pro-Motion see Section 5.2, "CAN Communications."

4.2.3.1   Connections & Associated Signals

The DK58113 board has a dual RJ45 connector (J6, J7) to allow daisy-chaining of DK58113 boards in a CANbus network. All pins in each connector are connected to the corresponding pin in the other connector.

Termination at each end of the cable run is recommended. ISO-11898 requires 120 Ohm termination at each end of the bus. Standard UTP (unshielded twisted pair) CAT5 Ethernet cabling can be used in most CAN applications. For added noise immunity, shielded cable can be used. Note that it is up to the customer to verify their network topology and operating parameters.

Here are the pinouts for both the J6 and J7 CAN connectors:

Signal Name	Pin #	Description
		J6, J7 – CAN Connectors
CAN+	1	Positive CAN signal connection
CAN-	2	Negative CAN signal connection
GND	3	Ground
No connect	4	Pass-through signal
No connect	5	Pass-through signal
No connect	6	Pass-through signal
GND	7	Ground
No connect	8	Pass-through signal

4.3   On-Board Switching Motor Amplifier

The DK58113 board contains a high-efficiency MOSFET power stage with PWM input control and leg current feedback. A different configuration is used for each motor type:

- Brushless DC motors are driven in a 3-phase bridge configuration consisting of 6 MOSFETs and 3 leg current sensors

- DC Brush motors are driven in an H-Bridge configuration consisting of 4 MOSFETs and 2 leg current sensors

- Step motors are driven with two H-Bridges, one for each phase, for a total of 8 MOSFETs and 4 leg current sensors

To operate the DK58113's on-board amplifier the JP1 and JP2 jumpers are installed in the 1-2 position, which is the factory default setting. The operating voltage range of the on-board amplifier is 12-56 volts, input at the J1 HV Power Connector.

All DK58113 on-board amplifier functions are controlled by the MC58113 IC. For complete information on the amplifier bridge control and current sense input functions of the MC58113 family ICs see the *Magellan Motion Control IC User Guide.*

4.3.1   Brushless DC Motor Drive

Figure 4-2 shows the arrangement of the DK58113 board's amplifier stage when the MC53113 IC is used or when the MC58113 IC is used with the Brushless DC motor type selected.
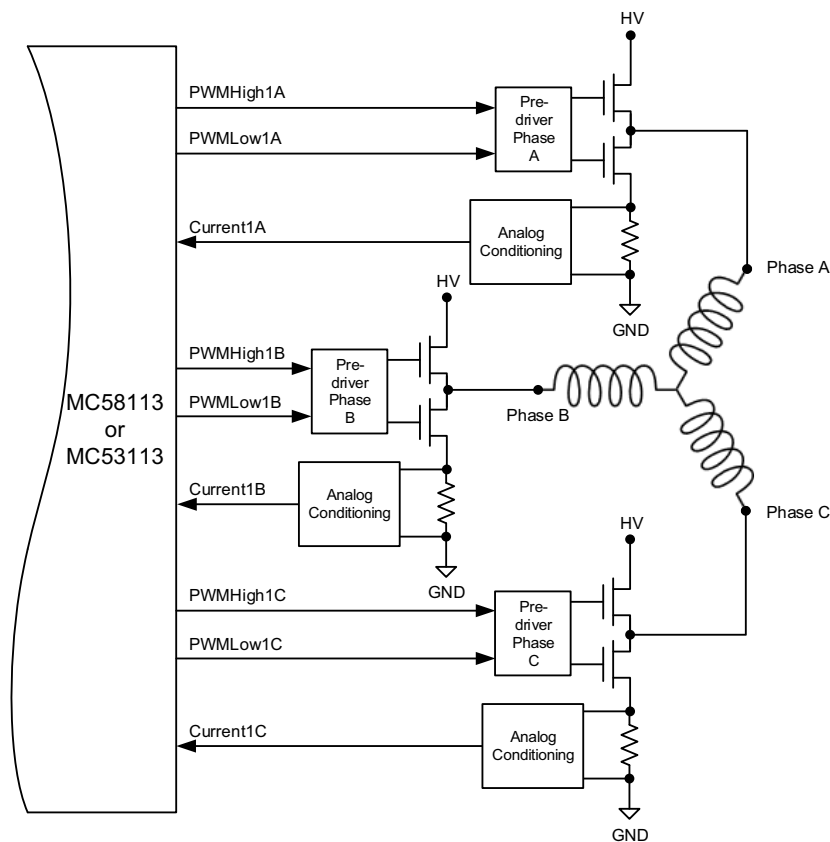
Figure 4-2: Brushless DC Motor Bridge Configuration

As shown in the table below six PWM output signals and three analog feedback signals interface between the MC58113 IC and the DK58113 board's switching amplifier.

MC58113 Signal	Description
PWMHigh1A	Digital high side drive output for motor phase A
PWMLow1A	Digital low side drive output for motor phase A
PWMHigh1B	Digital high side drive output for motor phase B
PWMLow1B	Digital low side drive output for motor phase B
PWMHigh1C	Digital high side drive output for motor phase C
PWMLow1C	Digital low side drive output for motor phase C
Current1A	Analog input containing the current flow through the low side of the switching bridge for phase A.

MC58113 Signal	Description
Current1B	Analog input containing the current flow through the low side of the switching bridge for phase B.
Current1C	Analog input containing the current flow through the low side of the switching bridge for phase C.

4.3.2 DC Brush Motor Drive

Figure 4-3 shows the arrangement of the DK58113 board's amplifier stage when the 51113 IC is used, or when the MC58113 IC is used with the DC Brush motor type selected.

As shown in the table below four PWM output signals and two analog feedback signals interface between the MC58113 IC and the DK58113 board's switching amplifier.
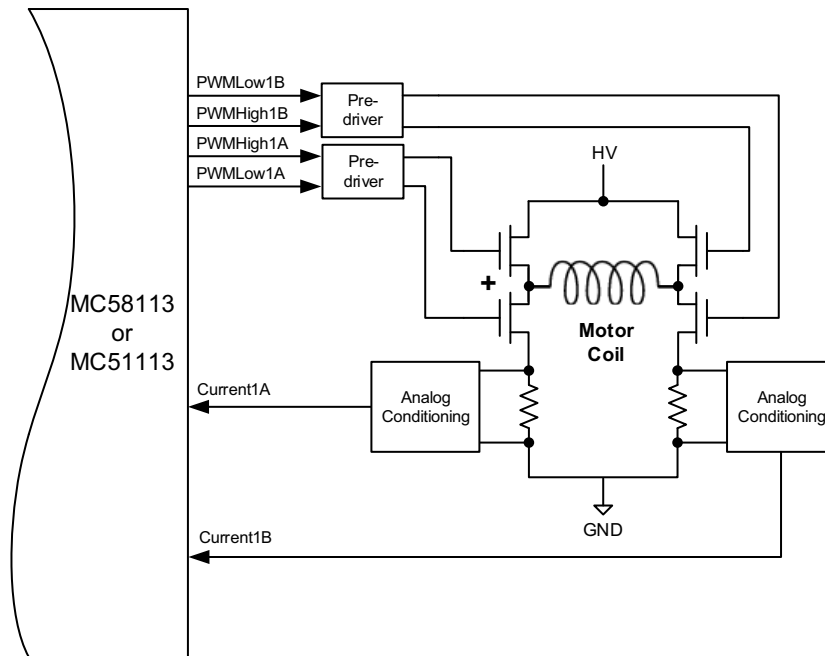
MC58113 Signal	Description
PWMHigh1A	Digital high side drive output for positive coil terminal
PWMLow1A	Digital low side drive output for positive coil terminal
PWMHigh1B	Digital high side drive output for negative coil terminal
PWMLow1B	Digital low side drive output for negative coil terminal
Current1A	Analog input containing the current flow through the positive leg of the bridge
Current1B	Analog input containing the current flow through the negative leg of the bridge

4.3.3 Step Motor Drive

Figure 4-4 shows the arrangement of the DK58113 board's amplifier stage when the MC54113 IC is used, or when the MC58113 IC is used with the two-phase step motor type selected.
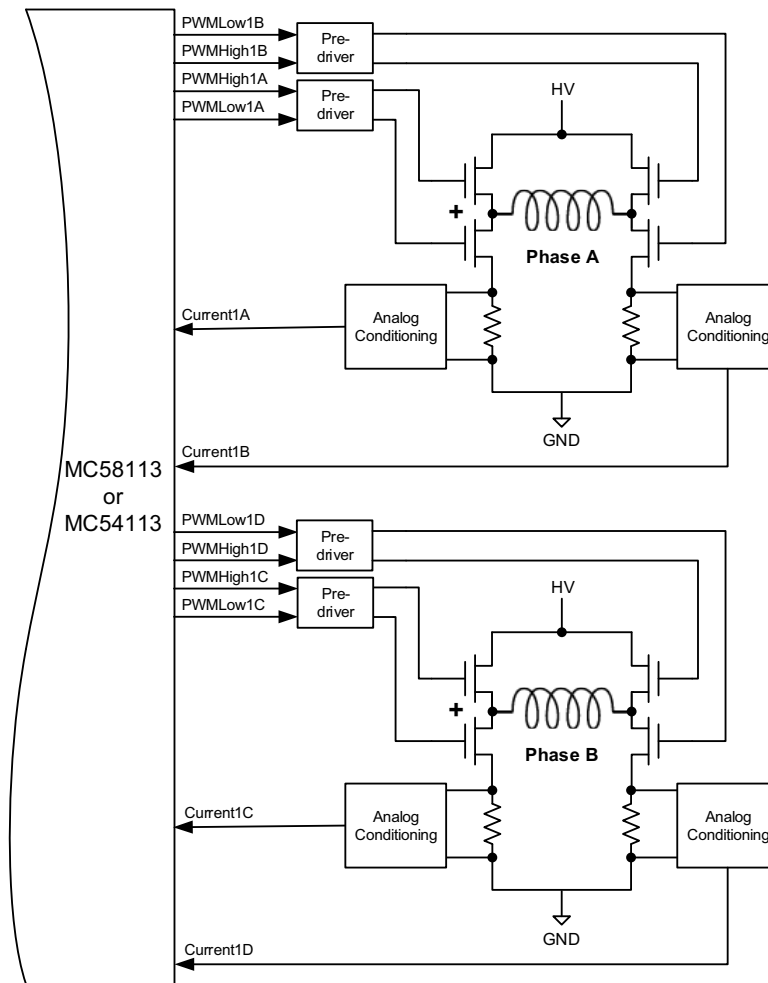
As shown in the table below eight PWM output signals and four analog feedback signals interface between the MC58113 IC and the DK58113 board's switching amplifier.

MC58113 Signal	Description
PWMHigh1A	Digital high side drive output for motor phase A, positive coil terminal
PWMLow1A	Digital low side drive output for motor phase A, positive coil terminal
PWMHigh1B	Digital high side drive output for motor phase A, negative coil terminal
PWMLow1B	Digital low side drive output for motor phase A, negative coil terminal
PWMHigh1C	Digital high side drive output for motor phase B, positive coil terminal
PWMLow1C	Digital low side drive output for motor phase B, positive coil terminal
PWMHigh1D	Digital high side drive output for motor phase B, negative coil terminal
PWMLow1D	Digital low side drive output for motor phase B, negative coil terminal
Current1A	Analog input containing the current flow through the positive leg of phase A bridge
Current1B	Analog input containing the current flow through the negative leg of phase A bridge
Current1C	Analog input containing the current flow through the positive leg of phase B bridge
Current1D	Analog input containing the current flow through the negative leg of phase B bridge

4.3.4   Amplifier-Related Settings

There are a number of MC58113 IC settings which control the DK58113 on-board switching amplifier and related current sense circuitry. The following table shows these settings:

Parameter	Value & Units
Motor Output Mode	PWM High/Low
PWM Dead Time	540 nSec
PWM Refresh Time	2,000 nSec
PWM Refresh Period	8 cycles
PWM Signal Sense	Active High
Minimum Current Read Time	2,000 nSec
Leg Current Conversion	.733 mA/count

When the DK58113 board is initialized with the Pro-Motion Axis Wizard these amplifier control and safety settings are programmed automatically. A description of running the Pro-Motion Axis Wizard can be found in Section 2.4, "Step #4 — First Time System Verification."

In addition to the above MC58113 amplifier control settings for the DK58113's on-board amplifier, the following table shows amplifier control settings that are application-specific and can be changed by the user:

Parameter	Default Value & Units	Comment
PWM Switching Frequency	20 kHz	This setting is motor-specific. Higher inductance motors should be set for 20 kHz. Lower inductance motors may use 40 or 80 kHz to reduce current ripple and minimize heat generation.

4.4   Drive Protection and Control Signals

All DK58113 board drive protection functions are controlled by the MC58113 IC. For complete information on the drive protection functions of the MC58113 family ICs see the *Magellan Motion Control IC User Guide*.

4.4.1   I^{2}t Current Foldback Protection

The DK58113 board provides I^{2}t current limiting. This feature protects the board amplifier by controlling its ability to operate above specific selected continuous current ratings.

When the current loop is enabled and the I^{2}t energy limit is exceeded, the phase currents are automatically folded back to a user programmable continuous current limit value. Alternatively, the DK58113 board can be configured to fault and disable the output stage when the I^{2}t energy limit is exceeded.

The following I^{2}t limits are the maximum allowed settings for the DK58113's on-board amplifier. When driving motors that have current and energy limits lower than these however the user may specify lower values.

Parameter	Value & Units
Brushless DC motor: Foldback Continuous Current Limit	5.0 A
Brushless DC motor: Foldback Total Energy Limit	125 A^{2}sec
DC Brush motor: Foldback Continuous DC Current Limit	5.0 A
DC Brush motor: Foldback Total Energy Limit	125 A^{2}sec
Step motor: Foldback Continuous Current Limit	5.0 A
Step motor: Foldback Maximum Energy Limit	125 A^{2}sec

4.4.2   Overtemperature Protection

The DK58113 board uses a temperature sensor to continuously monitor the temperature in the vicinity of the on-board power MOSFETs. The primary purpose of this function is to shut down motor output if the temperature limit is exceeded, thereby protecting the amplifier circuitry. The temperature may also be traced and viewed on Pro-Motion's Scope window display to understand how particular trajectory moves or amplifier settings affect the amplifier temperature.

Using Pro-Motion conversion of the MC58113 IC's ADC (analog to digital converter) reading from the DK58113 board's temperature sensor into degrees C is handled automatically. For user-designed boards however having Pro-Motion correctly display the temperature in degrees C is likely to require the user to construct a conversion table. For information on how to create such a table file refer to Appendix B, "Temperature Conversion Tables." The following temperature value is the maximum allowed setting for the DK58113 board.

Parameter	Value & Units
Temperature limit	75.0 C

The DK58113's temperature sensor is located on the DK58113 board and therefore will only function correctly when the on-board amplifier is used.

When the DK58113 board is initialized with the Pro-Motion Axis Wizard the required settings described in this section are programmed automatically by Pro-Motion. If parameters are settable the user is queried to change their default values if desired.

4.5   DC Bus

All DK58113 board DC Bus monitoring functions are controlled by the MC58113 IC. For complete information on the DC Bus monitoring functions of the MC58113 family ICs see the *Magellan Motion Control IC User Guide.*

Figure 4-5:
DC Bus
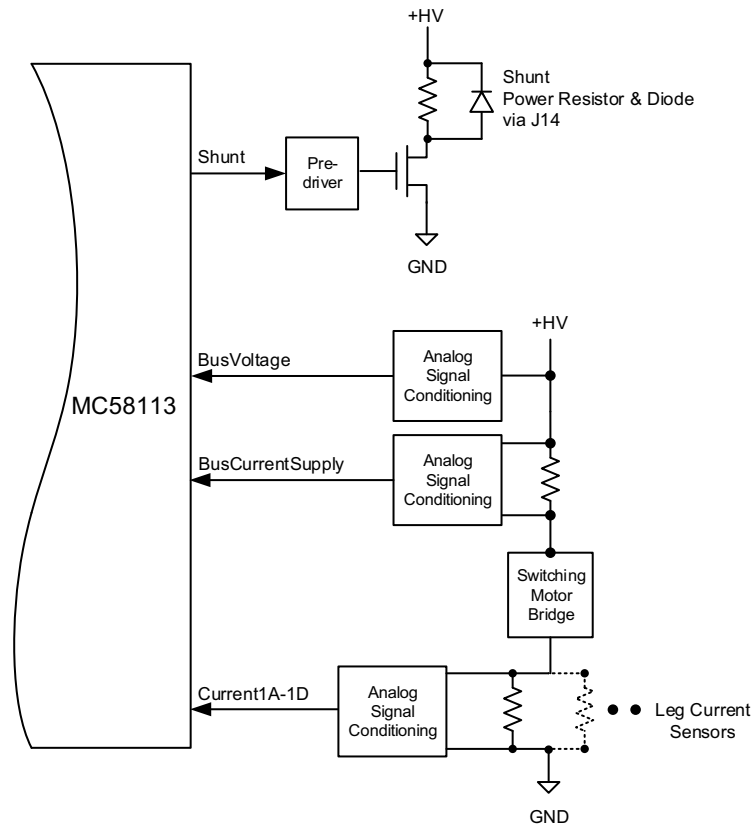**Monitoring**
Circuitry

Figure 4-5 shows the DC bus monitoring circuitry used with the DK58113 board. This circuitry is designed to work with the MC58113's DC bus management and protection functions. These functions include overcurrent protection, over and under voltage detection, and shunt resistor control.

4.5.1   DC Bus Current Monitoring

The DK58113 board monitors both the positive and negative DC bus current to detect overcurrent conditions including: line-to-line, line-to-power supply, and line-to-ground short circuits.

The following table shows the DC bus current monitoring scale values for the DK58113 board:

Parameter	Value & Units
Bus current supply conversion	.505 mA/count
Leg current return conversion	.733 mA/count

The following bus current limits are the maximum allowed settings for the DK58113 board:

Parameter	Value & Units
Bus Current Supply Limit	20.0 A
Bus Current Return Limit	20.0 A

Note that these values are designed to protect the DK58113 board circuitry. The user may specify lower values to protect the motor.

> DC bus current monitoring functions will not work when a remote amplifier is used.

4.5.2 DC Bus Voltage Monitoring

The DK58113 board monitors the main DC bus voltage (HV) for overvoltage and undervoltage conditions. These thresholds are user-settable within the DC Bus Voltage limits indicated below. DC bus voltage monitoring may operate even when a remote amplifier or a pulse & direction amplifier is used, as long as the +HV supply is still connected to the DK58113 board.

The following table shows the allowed MC58113 DC bus voltage limits for the DK58113 board:

Parameter	Value & Units
Undervoltage Limit	10.0 V
Overvoltage Limit	60.0 V

The following table shows the DK58113 board DC bus voltage scale factor.

Parameter	Value & Units
Bus Voltage Display	.966 mV/count

> When the DK58113 board is initialized with the Pro-Motion Axis Wizard the required settings described in this section are programmed automatically by Pro-Motion. If parameters are settable the user is queried to change their default values if desired.

4.6 Operating With External Amplifiers

In addition to having the MC58113 IC operate the DK58113's on-board switching amplifier, there are three external amplifier options that the DK58113 board supports; a remote switching amplifier, a PMD Atlas Amplifier, and a pulse & direction input amplifier. The subsequent sections describe how to configure and connect the DK58113 board to operate with these three external amplifier types.

4.6.1 Operating a Remote Switching Amplifier

To operate the DK58113 board with a remote switching amplifier jumpers JP1 and JP2 must be set to the 2-3 position, thereby disabling the DK58113's on-board amplifier.

Figure 4-6: DK58113 Board to Remote Amplifier Connections
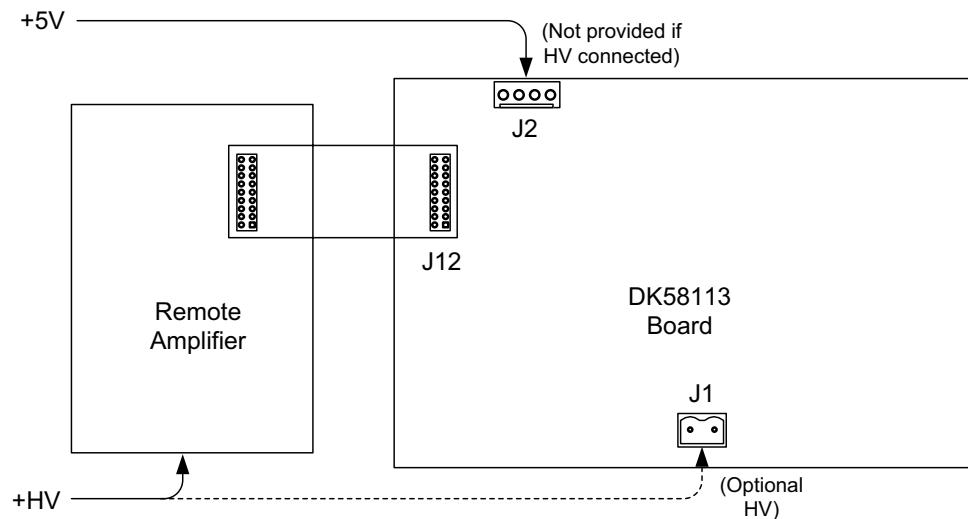
The DK58113 board supports the ability to drive a remote switching amplifier via its J12 connector. Development of a user-designed amplifier connected directly to the DK58113 board may be useful to service higher power motors, or motors or actuators that require special bridge configurations.

Connections for a remote amplifier are shown in Figure 4-6. Note that connection of the remote amplifier's HV supply to the DK58113 board is optional, but recommended as long as HV is within the DK58113 board's input range which is 12-56V. Doing so allows use of the DK58113 board's over and undervoltage check feature. If you connect HV to the DK58113 board you should not provide 5V via the J2 connector, and conversely if you do not provide HV to the DK58113 board 5V needs to be provided at J2.

> Operating the DK58113 board with a remote switching amplifier requires that jumpers JP1 and JP2 be set to the 2-3 position, thereby disabling the DK58113's on-board amplifier. Failure to do so may result in damage to the DK58113 board, the remote amplifier, and the controlled motor.

4.6.1.1   Supported Features

Central to the design of a switching amplifier is the choice of MC58113 motor output mode. There are several different signaling and functional modes supported including PWM High/Low (which is the mode the DK58113 board's amplifier uses), 50/50 PWM and sign/magnitude PWM. Choice of motor output mode depends on whether your design will provide current control and on the signal interface scheme of the bridge circuitry used.

Not all of the features associated with the DK58113 on-board amplifier are supported when a remote amplifier is operated. This is shown in the table below:

Feature	Supported?	Comments
PWM signal generation	Yes	Supported for all PWM modes
Analog current feedback	Yes	PWM High/Low control mode only
I2t	Yes	PWM High/Low control mode only, control settings are amplifier design-specific
Overtemperature protection	No	
Supply overcurrent detect	No	
Return overcurrent detect	No	

Feature	Supported?	Comments
DC Bus overvoltage detect	Yes	Supported when HV is wired to DK58113 board's J1 connector
DC Bus undervoltage detect	Yes	Supported when HV is wired to DK58113 board's J1 connector
Shunt Control	No	

4.6.1.2   Control Settings and Bring-Up

When a remote amplifier circuit is ready for testing the MC58113 family IC that will drive it must be programmed with the correct control settings. The table below provides a list of amplifier control settings when the PWM High/Low output mode is used. This is the MC58113 IC motor output mode that provides current control:

Setting	Comments
PWM Switching Frequency	This setting is motor-specific. Higher inductance motors should be set for 20 kHz. Lower inductance motors may use 40 or 80 kHz to reduce current ripple and minimize heat generation.
PWM Dead Time PWM Refresh Time PWM Refresh Period PWM Signal Sense Minimum Current Read Timing	These settings are amplifier design specific and vital to safe and correct operation of the amplifier bridge.
Continuous Current Limit Foldback Energy Limit	Settings are typically different for different motor types driven (Brushless DC, DC Brush, step motor).
Overvoltage Limit Undervoltage Limit	Can be used with all PWM output modes if HV is connected at J1.
Leg Current Scale	Specifies relationship between ADC current input reading to actual current in amps used by Pro-Motion to correctly display current in amps.

To bring up an untested remote amplifier design Pro-Motion can be used for programming the above settings, commanding the amplifier manually, and creating scope displays of the amplifier function.

Once the critical switch control settings have been entered such as PWM dead time, PWM signal sense, etc… a typical next step is to set the MC58113's Operating Mode to Motor Output only. This is done using the Operating Mode box of Pro-Motion's Axis Control window. Enable Motor Output and disable all higher level functions such as current loop, position loop, etc…

The Motor Control box in the Axis Control window allows you to specify specific PWM duty cycle settings and then check with an oscilloscope whether the signal output from the switcher appears as expected. Be aware that multi-phase motors such as step motors and Brushless DC motors will distribute a given specified motor command to multiple phases. To determine the exact command values sent to each phase Pro-Motion's scope trace function can be used.

As you build up more layers of amplifier function you can eventually enable the current loop by checking the corresponding radio button in the Operating Mode box and by setting gain values using the Current Loop box of the Axis Control window. Proper current loop function can be followed by enabling position loop function and trajectory generation until the motion control functionality using the remote amplifier is complete.

> Pro-Motion Axis Wizard should not be used when initializing the control settings of a remote amplifier. Axis Wizard assumes that you are using the on-board amplifier and therefore may program parameter values not appropriate for a user-designed remote amplifier. To set these parameters use the Pro-Motion Axis Control window.

### 4.6.1.3   Connections and Associated Signals

All of the signals used to operate a remote amplifier are connected directly to the MC58113 IC without buffering or other processing circuitry. For more information on the control options the MC58113 Motion Control IC provides and details of signal level and signal function refer to the *Magellan Motion Control IC User Guide* and the *MC58113 Electical Specifications*.

The table below shows the signals associated with operation of a remote amplifier:

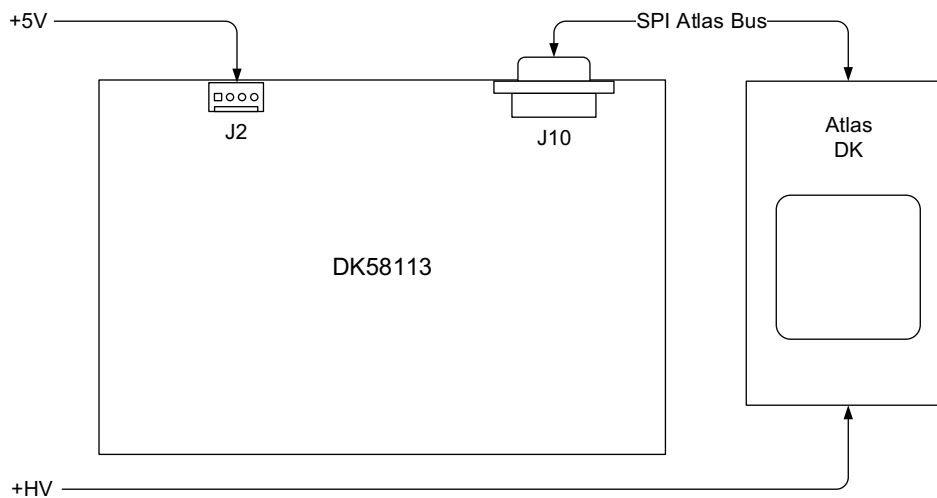| Signal Name	Pin #	Description
J12 – Remote Amplifier Connector		
AmplifierEnable1	1	AmplifierEnable1 output signal from MC58113
PWMOutputDisable	2	PWMOutputDisable input signal to MC58113
PWMHigh1A	3	PWMHigh1A output signal from MC58113
PWMLow1A	4	PWMLow1A output signal from MC58113
PWMHigh1B	5	PWMHigh1B output signal from MC58113
PWMLow1B	6	PWMLow1B output signal from MC58113
PWMHigh1C	7	PWMHigh1C output signal from MC58113
PWMLow1C	8	PWMLow1C output signal from MC58113
PWMHigh1D	9	PWMHigh1D output signal from MC58113
PWMLow1D	10	PWMLow1D output signal from MC58113
GND	11	Ground
AGND	12	Analog Ground
Current1A	13	Current1A analog input signal to MC58113
Current1B	14	Current1B analog input signal to MC58113
Current1C	15	Current1C analog input signal to MC58113
Current1D	16	Current1D analog input signal to MC58113

4.6.2   Operating an Atlas Amplifier

To operate the DK58113 board with an Atlas amplifier jumpers JP1 and JP2 must be set to the 2-3 position, thereby disabling the DK58113's on-board amplifier.
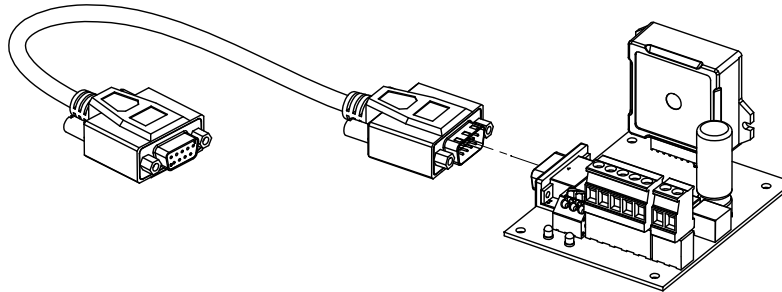
Connector J10 provides a DB-9 connection that is compatible with PMD's Atlas Digital Amplifier single-axis developer kit. Any Atlas motor type (DC Brush, Brushless DC, or step motor) can be driven.

When installing the Atlas DK with the DK58113 board HV power should be provided to the Atlas DK at its J1 HV connection, with power ground connecting at its J1 Pwr_Gnd connection. As shown in the diagram above only the Atlas DK should receive the HV voltage, while the DK58113 board should be powered with 5 volts at J2.

Operating the DK58113 board when using an Atlas amplifier is more or less identical to operating with the on-board amplifier. The motor to be controlled in this configuration should be initialized with the Pro-Motion Axis Wizard, with any required control settings automatically handled by Pro-Motion.

Operating the DK58113 board with an Atlas Amplifier requires that jumpers JP1 and JP2 be set to the 2-3 position, thereby disabling the DK58113's on-board amplifier. Failure to do so may result in damage to the DK58113 board, the Atlas amplifier, or the controlled motor.
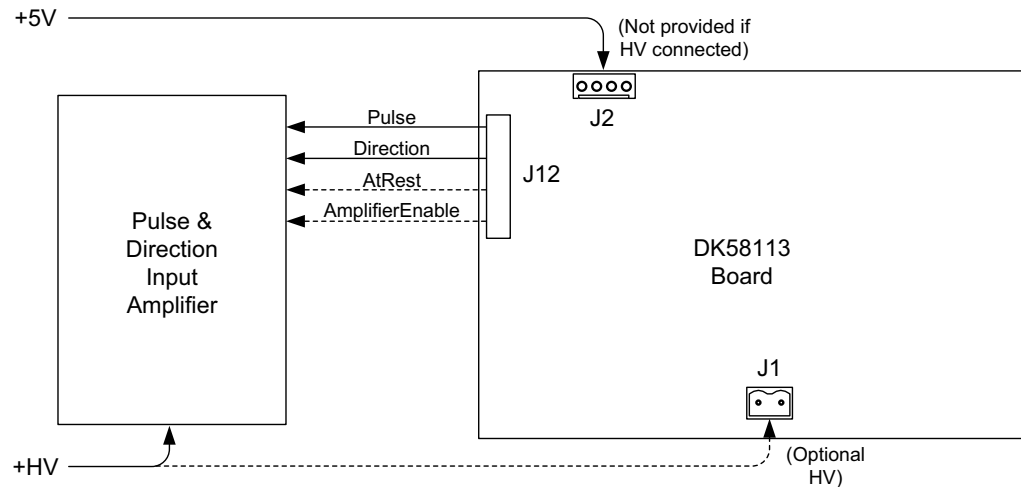
4.6.3   Operating a Pulse & Direction Amplifier

To operate the DK58113 board with a pulse & direction amplifier jumpers JP1 and JP2 must be set to the 2-3 position, thereby disabling the DK58113's on-board amplifier.

The DK58113 board supports the ability to drive a pulse & direction amplifier via its J12 Connector. Four signals are output that may be useful in driving a pulse & direction amplifier; Pulse, Direction, AtRest, and AmplifierEnable. Note that while the pulse & direction signal format is most commonly used to drive step motors, it can be input into any external amplifier that accepts this position command format.

Figure 4-9: DK58113 Board to Pulse & Direction Input Amplifier Connections

Power connections for a pulse & direction amplifier are shown in Figure 4-9. Note that connection of the amplifier's HV supply to the DK58113 board is optional, but recommended as long as HV is within the DK58113 board's input range which is 12-56V. Doing so allows use of the DK58113 board's over and undervoltage check feature. If you connect HV to the DK58113 board you should not provide 5V via the J2 connector, and conversely if you do not provide HV to the DK58113 board 5V needs to be provided at J2.

The pulse & direction amplifier to be controlled in this configuration should be initialized using the Pro-Motion Axis Wizard, with any control setting options automatically handled by Pro-Motion.

> Operating the DK58113 board with a pulse & direction amplifier requires that jumpers JP1 and JP2 be set to the 2-3 position, thereby disabling the DK58113's on-board amplifier. Failure to do so may result in damage to the DK58113 board and in unexpected behavior in the pulse & direction input amplifier.

4.6.3.1   Connections and Associated Signals

All of the signals used to operate a pulse & direction amplifier are connected directly to the MC58113 IC without buffering or other processing circuitry. For more information refer to the *Magellan Motion Control IC User Guide* and the *MC58113 Electrical Specifications*.

The table below shows the signals associated with operation of a pulse & direction amplifier with the DK58113 board:

Signal Name	Pin #	Description
J12 – Remote Amplifier Connector		
AmplifierEnable1*	1	Amplifier enable output signal from MC58113. The conditions by which this signal goes active are programmable but are typically used to disable the amplifier output when a stall is detected or to control the timing of when the amplifier is enabled during system initialization.
Pulse1	5	PWMHigh1B/Pulse1 signal from MC58113
Direction1	6	PWMLow1B/Direction1 signal from MC58113
AtRest*	7	PWMHigh1C/AtRest signal from MC58113
GND	11	Ground

These signals are optional and not supported by all pulse & direction input amplifiers.

> The Pulse, Direction, AtRest, and AmplifierEnable output signals are single-ended (non-differential) 3.3V digital outputs, and therefore some cable-connected amplifiers may not work reliably without the addition of signal processing circuitry for these signals. Check the specifications of the amplifier you are using to determine its compatibility with direct connection to these signals.

4.7   Shunt Regulation

The DK58113 board's J14 connector provides a connection for a shunt resistor and diode that may be used to regulate overvoltage conditions on the DC bus. Such conditions can occur during deceleration of a motor with a large inertia. As shown in Figure 4-5 the MC58113 provides a shunt PWM output, which in turn drives a MOSFET switch on the DK58113 board.

The resistor connected at J14 should have a resistance such that the current flow through the Shunt MOSFET does not exceed 10 amps. For example with an HV supply of 48 Volts, this means a resistance of no less than 4.8 ohms. The diode, which is connected in parallel to the resistor, should have a voltage and current rating at least equal to those of the MOSFET. For the DK58113 this means a voltage and current rating of 100 volts and 10 amps or higher.

For applications below these limits, the actual resistance used is application specific and depends on the nature of the anticipated over voltage generating conditions, the power supply used, and the wattage rating of the resistor. The output duty cycle of the shunt bypass can be specified by the user.

Example: a shunt resistor with a resistance of 10 ohms is installed and a comparison value of 51 Volts and a PWM duty cycle of 75% are specified by the user. When the +HV voltage exceeds 51.0 Volts, HV will be connected to GND via the shunt resistor resulting in an effective average current flow of (51.0V * .75)/10 ohms = 3.825 amps.

The default value of the shunt comparison mechanism is disabled. To enable, both a voltage comparison value and a PWM output duty cycle are specified. This can be accomplished by selecting the Drive Safety box in Pro-Motion's Axis Control window. For a complete application note on DC Bus voltage regulation including shunt regulation see Section 3.11.4, "Application Note – Monitoring and Enhancing DC Bus Voltage Stability."
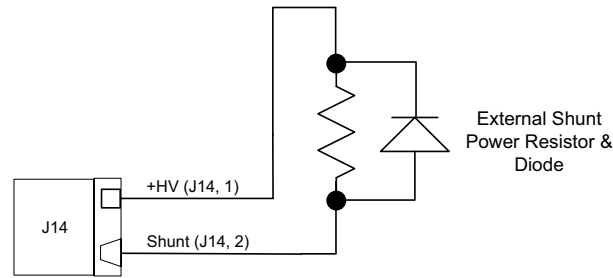
> It is the responsibility of the user to connect and to specify a shunt-related resistor and diode that are safe for the application being controlled.

4.7.1   Connections and Associated Signals

The shunt resistor and diode should be wired as shown in the diagram below.

The table below shows the signals associated with the shunt function:

Signal Name	Pin #	Description
		J14 – Shunt Connector
HV	I	+HV
Shunt	2	Switched connection to ground

4.8   Motion Peripheral Signals

All of the motion peripheral signals described in this section are controlled by the MC58113 IC. For complete information on the motion peripheral-related functions of the MC58113 family ICs see the *Magellan Motion Control IC User Guide.*

4.8.1   Quad and Index Signal Inputs

The DK58113 board supports two full channels of quadrature and index signal input with count rates up to 25 Mcounts per second, the primary axis (#1) and auxiliary axis (#2). These signals may alternatively input pulse and direction signals to provide position input.

The differential input circuitry for the encoder A, B and Index signals is shown in Figure 4-11. This circuit accepts both differential and single-ended signals in the range of $0 - 5$ V. For single-ended operation, the unused input should be left floating. Index capture occurs upon a low to high transition, however this signal sense interpretation can be user specified. Index capture is not qualified with the quad A and quad B signals.
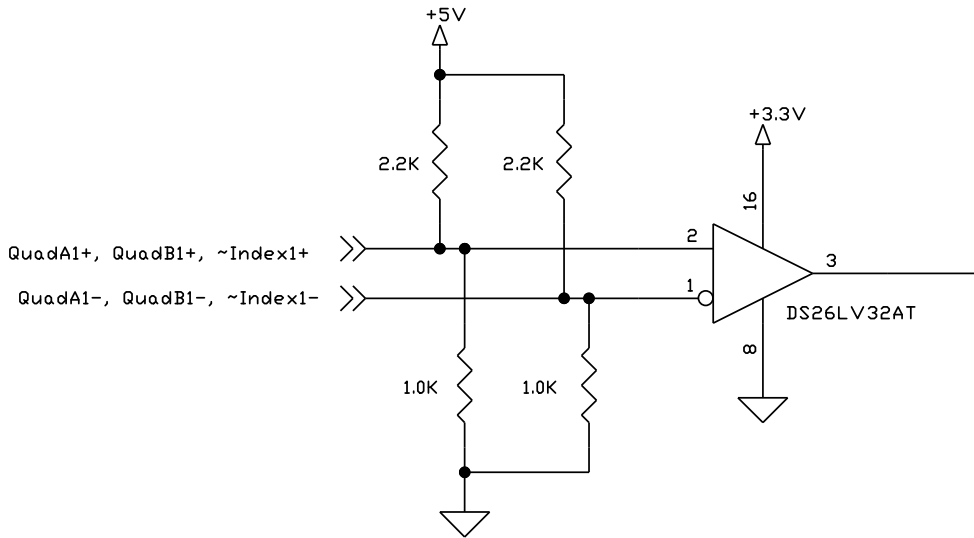
**Figure 4-11:
Main Encoder
Input Circuits**

Limit switch signal inputs are active when low. Home signal inputs are active when high. Both of these signal interpretations can be user-specified however.

4.8.2   Hall Inputs

The input buffer for the Hall A, B and C signals is shown in Figure 4-12. This circuit accepts signals in the range of $0 - 24$ V and has TTL compatible, Schmitt trigger thresholds. It has a pull-up to 5V to allow direct interfacing to open collector sources without the need for an external pull-up resistor and an R-C low pass filter to reject noise.
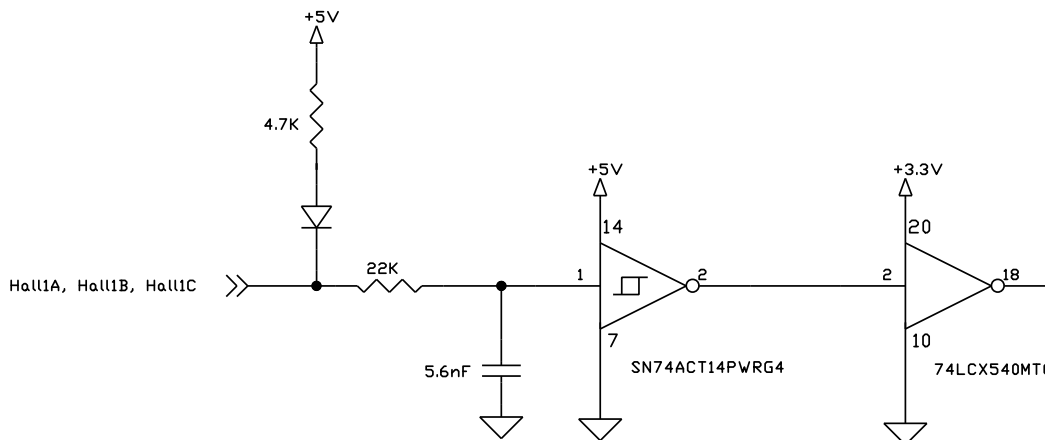
**Figure 4-12:
Hall Input
Circuits**

Hall signals are only used with Brushless DC motors. They are used to directly commutate the motor in 6-step commutation mode or to provide an absolute phase reference for sinusoidal commutation.

4.8.3   Limit and Home Inputs

The input buffer for end-of-travel Limit and Home signals is shown in Figure 4-13. This circuit accepts signals in the range of 0–24 V and has TTL compatible, Schmitt trigger thresholds. It has a pull-up to 5V to allow direct interfacing

to open collector sources without the need for an external pull-up resistor and a 1.3 kHz R-C low pass filter to reject noise.
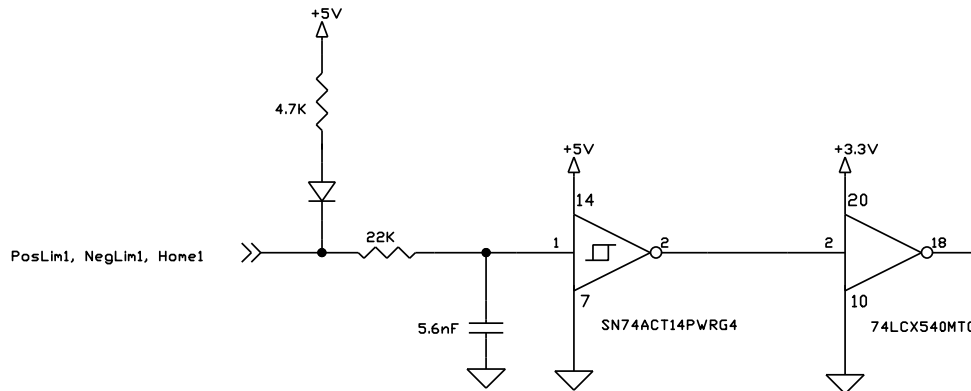
Figure 4-13: Limit and Home Input Circuits

4.8.4   AxisIn and AxisOut Signals

The input buffer for the AxisIn signal is shown in Figure 4-14. This circuit accepts signals in the range of 0 – 24 V and has TTL compatible, Schmitt trigger thresholds. It has a pull-up to 5V to allow direct interfacing to open collector sources without the need for an external pull-up resistor and a 13 kHz R-C low pass filter to reject noise.
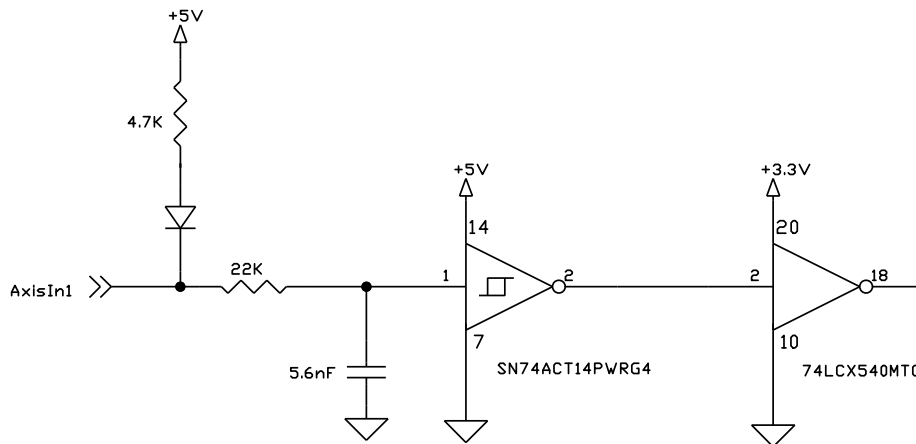
Figure 4-14: AxisIn Circuit

The output driver for the AxisOut signal is shown in Figure 4-15. This circuit can continuously sink over 100 mA and source 4mA from a pull-up resistor to 5V. The diode in series with the pull-up resistor allows loads powered from up to 24 VDC to be switched. The FET driver is internally protected from shorts up to 30 V.
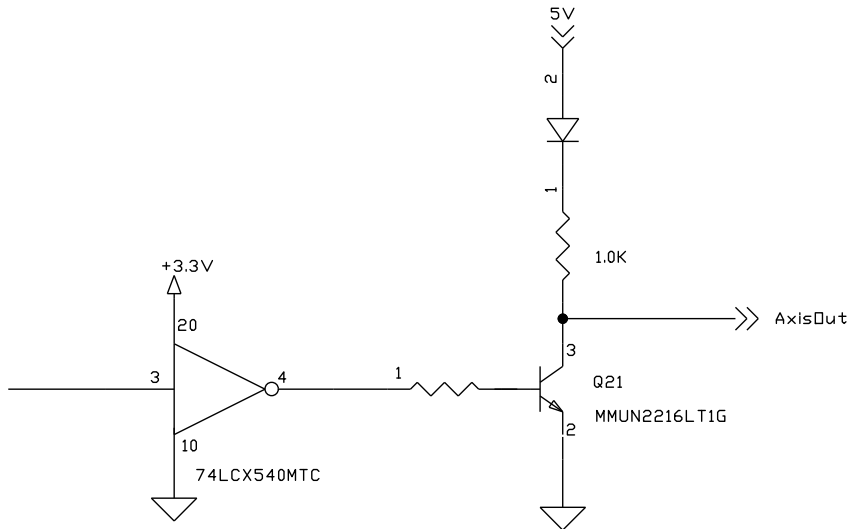
Figure 4-15: AxisOut Circuit

AxisIn and AxisOut are general purpose I/O signals. They are not dedicated to any particular motion control function but can be programmed to implement a wide array of system integration functions. See the *Magellan Motion Control IC User Guide* for more information on configuring and programming these signals.

4.8.5 Connections and Associated Signals

The table below shows the signals associated with the above functions with the DK58113 board:

Signal Name	Pin #	Description
		J8 – Axis 1 Feedback Connector
QuadA1+/Pulse1+	1	Axis 1 quadrature A+ or Pulse+ signal input
QuadA1-/Pulse1-	2	Axis 1 quadrature A- or Pulse- signal input
QuadB1+/ Direction1+	3	Axis 1 quadrature B+ or Direction+ signal input
QuadB1-/Direction1-	4	Axis 1 quadrature B- or Direction- signal input
GND	5	Ground
Index1+	6	Axis 1 Index+ input
Index1-	7	Axis 1 Index- input
HallA	8	Axis 1 Hall A input
HallB	9	Axis 1 Hall B input
HallC	10	Axis 1 Hall C input
Home1	11	Axis 1 Home input
PosLim1	12	Axis 1 Positive direction limit switch input
NegLim1	13	Axis 1 Negative direction limit switch input
Vcc	14	+5V output

Signal Name	Pin #	Description
		J9 – Axis 2 Feedback Connector
QuadA2+/Pulse2+	1	Axis 2 quadrature A+ or Pulse+ signal input
QuadA2-/Pulse2-	2	Axis 2 quadrature A- or Pulse- signal input
QuadB2+/ Direction2+	3	Axis 2 quadrature B+ or Direction+ signal input

Signal Name	Pin #	Description
QuadB2-/Direction2-	4	Axis 2 quadrature B- or Direction- signal input
GND	5	Ground
Index2+	6	Axis 2 Index+ input
Index2-	7	Axis 2 Index- input
Home2	11	Axis 2 Home input
AxisIn1	12	Axis 1 AxisIn signal input
AxisOut1	13	Axis 1 AxisOut signal output
Vcc	14	+5V output
GND	15	Ground

# 4.9   Enable and FaultOut Signals

The Enable and FaultOut signals described in this section are controlled by the MC58113 IC. For complete information on these signal functions see the *Magellan Motion Control IC User Guide.*

These dedicated signals are typically used to implement a safety interlock between the DK58113 board and other control portions of the system. Enable is an active-low input that must be tied or driven low for the DK58113 output stage to be active.

FaultOut indicates a serious problem. When the DK58113 board is operating properly FaultOut is inactive. The polarity of these signals is fixed and cannot be changed via software.

The input buffer for the /Enable input is shown in Figure 4-16. This circuit accepts signals in the range of 0 – 24 V and has TTL compatible, Schmitt trigger thresholds. It has a pull-up to 5V to allow direct interfacing to open collector enable sources without the need for an external pull-up resistor and a 1.3 kHz R-C low pass filter to reject noise.

Figure 4-16: Enable Input Circuit
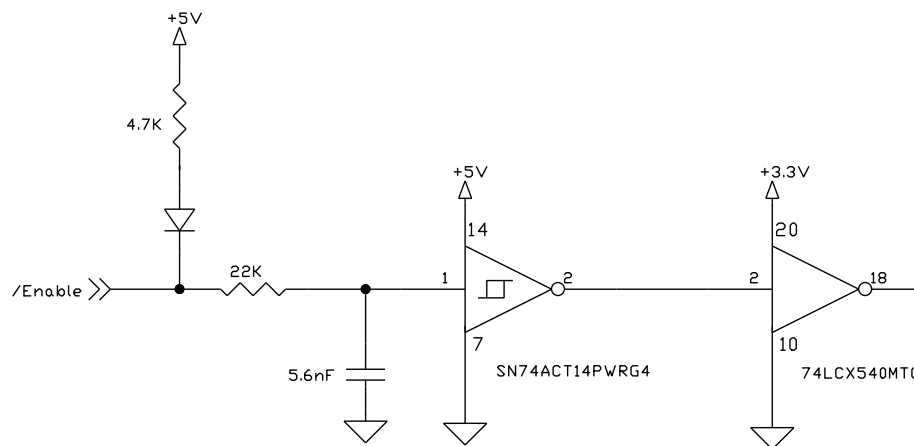
The output driver for FaultOut is shown in Figure 4-17. This circuit can continuously sink over 100 mA and source 4mA from a pull-up resistor to 5V. The diode in series with the pull-up resistor allows loads powered from up to 24 VDC to be switched. The FET driver is internally protected from shorts up to 30 V.
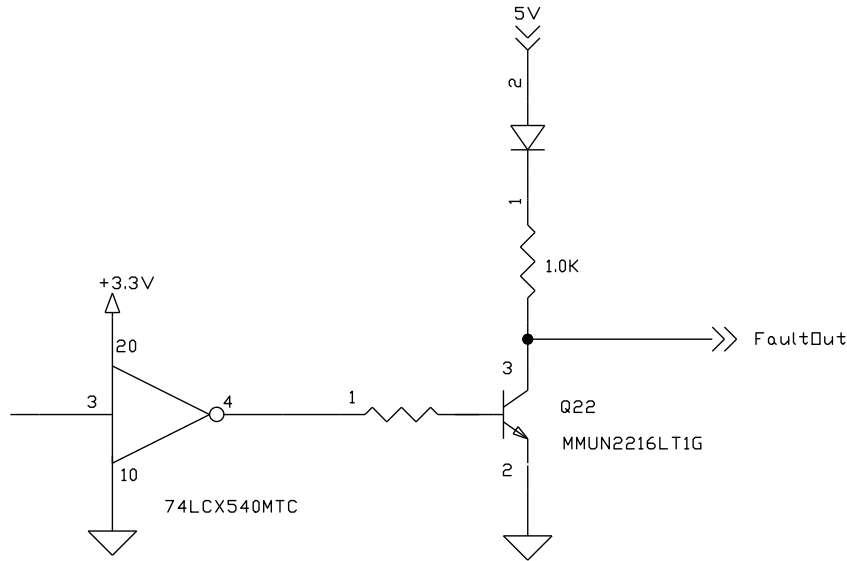
**Figure 4-17:
FaultOut
Circuit**

4.9.1 Connections and Associated Signals

The table below shows the signals associated with the Enable and FaultOut functions of the DK58113 board:

Signal Name	Pin #	Description
J4 – Amplifier Signal Connector		
Enable	1	Active low Enable digital input signal
FaultOut	2	Active High digital FaultOut output signal
GND	3	Ground

4.10 Multi-board Synchronization

The Synch Connector located on the DK58113 board (J13) allows for the synchronization of multiple DK58113 boards within a system. All DK58113 board synch functions are controlled by the MC58113 IC. For complete information on the synching functions of the MC58113 family ICs see the *Magellan Motion Control IC User Guide.*

To synchronize two or more DK58113 boards a cable is used wired in a "T" configuration as shown in .
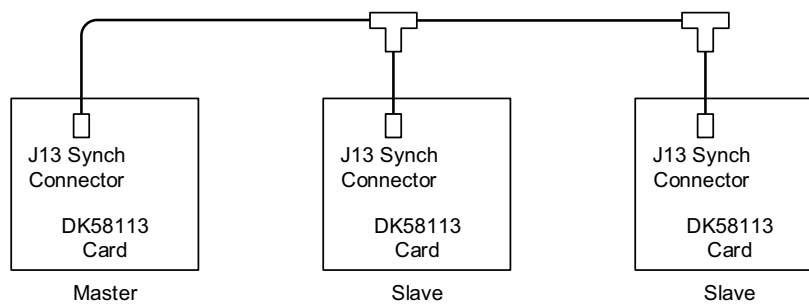
**Figure 4-18:
Synch I/O
Connector to
ThreeDK58113
Boards**

4.10.1 Connections and Associated Signals

Synch is a digital 3.3V signal connected directly to the MC58113 IC without buffering or other processing circuitry.

The table below shows the signals associated with operation of the inter-board synch function:

Signal Name	Pin #	Description
J13 – Synch Connector		
Synch	1	This pin inputs or outputs a synchronization signal that can be used to synchronize the loop rates of multiple MC58113s with each other or with another external source.
GND	2	Ground

4.11 MC58113 IC Reset

The Reset signal located on the J11 Host SPI Connector provides a hardware reset of the MC58113 family IC on the DK58113 board. While not needed for power-up initialization, an externally commanded hardware reset function may be useful in some applications. The external reset signal has active low signal function.

For information on the reset function of the MC58113 IC refer to the *MC58113 Electrical Specifications*.

4.11.1 Connections and Associated Signals

The reset signal is driven low by the MC58113 IC, therefore to exercise the external reset function the signal must be driven with an open-drain/open-collector output.

The table below shows the signals associated with the hardware reset function:

Signal Name	Pin #	Description
J11 – Host SPI Connector		
GND	7	Ground
Reset	8	Active low reset signal for the MC58113 IC. Must be driven with open-drain/open-collector output.

5. Communication Port Connections

In This Chapter

▶ Serial Communications
▶ CAN Communications
▶ SPI Communications

In this chapter we provide additional information on Pro-Motion communication connections including how to set up and operate PMD controllers for RS232, RS485, CAN, and SPI communications.

This information may be useful to change the connection mode originally set up in Chapter 2, *Quick Start Guide*, or to add additional PMD controllers to the existing Pro-Motion setup.

5.1   Serial Communications

DK58113 boards support RS232, RS422, and RS485 communications. For additional information on the DK58113 board's serial interfaces refer to Section 4.2.1, "RS232/485 ."

In the following sections we will detail how to connect from a PC to the PMD controller both for RS232 and RS485.

5.1.1   RS232 Communications
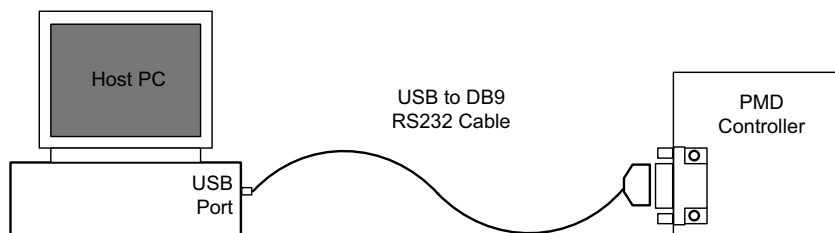
For the PC to communicate to the DK58113 board via RS232 you will use the USB to DB9 serial converter cable (PMD P/N Cable-USB-DB9) which is included with DK58113 developer kits. The DK58113 board directly accepts this cable at its J5 connector.

5.1.2   Changing the Active RS232 Settings

Here is the Pro-Motion sequence to change the active RS232 parameter settings in the PMD controller and in Pro-Motion.

1   With Pro-Motion communications via the RS232 port functioning properly, click the Device Control toolbar icon. The Device Control window appears.

2   Click Network I/O. The Network I/O Defaults dialog box appears.

3 Click the Serial tab. This window appears with data entry fields for the default serial port setting such as the baud rate. This is shown below with default values visible.
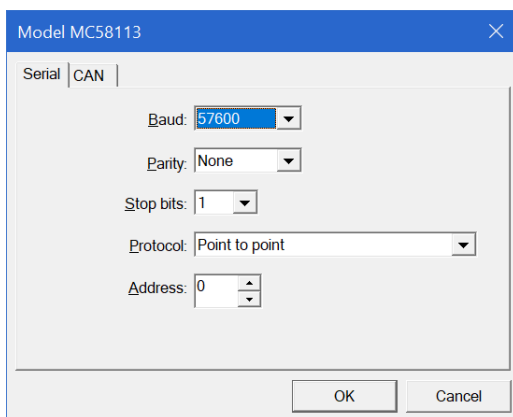
4 Enter the desired new comm settings in the corresponding data fields. The Protocol field should be set to point-to-point and the Address field can be left unchanged.

5 Click OK to set as the PMD controller's and Pro-Motion's active RS232 communication settings.

Note that these changes were made to the active settings and are not stored in the PMD Controller's NVRAM. Therefore upon a power-up or reset of the PMD controller it will revert to using its default values (or values that were stored in the motion IC's NVRAM).

5.1.3 Storing Communication Settings in NVRAM

Communication parameter settings, whether for RS232, RS485, or CAN, can be stored in the MC58113 IC's NVRAM using scripts, thereby allowing that device to come out of powerup or reset with specific programmed communication settings. Note that although MC58113 ICs support SPI, because they function as SPI slaves there are no communication settings needed in the motion IC.

The script line that provides the communication settings can be placed in a script by itself, or may be part of a larger script that also specified control settings such as PID gains and safety settings. The screen capture below shows a script with a single line entry that sets communication parameters. In this example the entry specifies serial communication parameters consisting of a baud rate of 115200, 8 data bits, no parity bit, 1 stop bit, and a protocol of point-to-point (RS232).
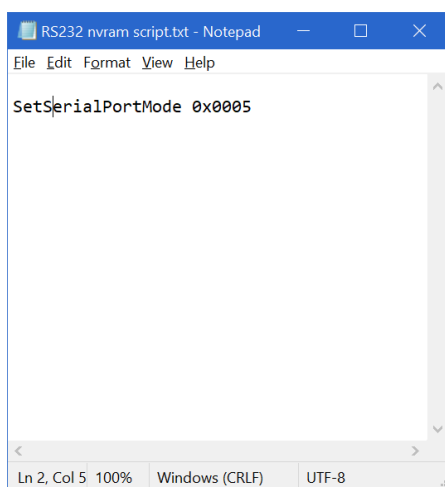
To download a script into the motion control IC's NVRAM the NVRAM button of Pro-Motion's Device Control window is used. For details see

5.1.4 RS232 NVRAM Settings

For RS232 changes to be permanently stored in the PMD controller these settings must be saved in the controller's NVRAM. For MC58113 ICs this is accomplished with Command window scripts. See Section 5.1.3, "Storing Communication Settings in NVRAM" for more information.

RS232 NVRAM settings are specified using a script with the command **SetSerialPortMode**. The complete format of this command is detailed in the *C-Motion Magellan Programming Reference*, but for convenience the table below provides script line entries to program various common RS232 settings.

Baud Rate	# Data Bits-Parity-#Stop bits	Protocol	Address	Corresponding Script Line
57,600	8-N-1	Point-to-point	N/A	SetSerialPortMode 0x0004
115,200	8-N-1	Point-to-point	N/A	SetSerialPortMode 0x0005
230,400	8-N-1	Point-to-point	N/A	SetSerialPortMode 0x0006
460,800	8-N-1	Point-to-point	N/A	SetSerialPortMode 0x0007

Here are the steps for storing RS232 settings into the PMD controller's NVRAM using scripts and activating these settings in the unit:

1 Compose a text script containing the command line that provides the desired RS232 settings. If not in the table above refer to the *C-Motion Magellan Programming Reference*.

2 With Pro-Motion communicating successfully to the PMD Controller via RS232, select the NVRAM button of the Pro-Motion Device Control window. Specify the file name to download, click Download!, and when the download is completed click Close

3 Disconnect existing RS232 communications using the Disconnect toolbar icon

4 Power down the PMD Controller, wait a few seconds, and repower the PMD Controller

> When changing the RS232 parameters special care should be taken since this is the channel being used to communicate from Pro-Motion to the PMD controller. If for some reason communication is lost and cannot be re-established after the RS232 settings are changed, it is recommended that you try to connect via CAN, if available, by setting Pro-Motion to communicate at the CAN default settings. For the default communication settings for CAN refer to Section 4.2.3, "CANbus."

5.1.5 Pro-Motion RS232 Settings

If the RS232 communication settings used by the PMD controller have been changed the settings used by Pro-Motion must be set to the same settings.

To set Pro-Motion's RS232 parameters:

1 With no RS232 connection to the PMD controller operating, click the Connect toolbar icon.

2 Select Serial from the list of options.

3 Select the comm port that the RS232 connection is located at and select OK.

4 Enter the same RS232 settings as were programmed into the PMD controller's NVRAM previously.

5 When complete, click OK.

If RS232 communication is successful a graphical icon representing the PMD controller on the network will be loaded into the Project window. If communication is not successful, a Communications Timeout Error dialog box appears. If this happens recheck the connections and retry to establish communications with the PMD controller unit.

With RS232 communications established, a useful optimization of the RS232 connection may be to reduce the USB latency between message packet sends. If this was already done for your PC during quick setup then there is no need to set this again. If not, the next paragraph describes how this can be done:

First, open the Windows Device Manager by typing "Device Manager" in the Windows search box. Next, find Ports (COM & LPT) from the list and click on it, then right-click the USB Serial Port (COMn) of the serial port you are using and then select Properties. Click on the Port Settings tab and then select the Advanced button. Change the field for Latency Timer (msec) to a value of 1. Click OK on both windows and close the Device Manager. The driver is now optimized for use with the machine controller drive.

5.1.6   RS485 Communications

Figure 5-2: Example Setup for Communications via RS485
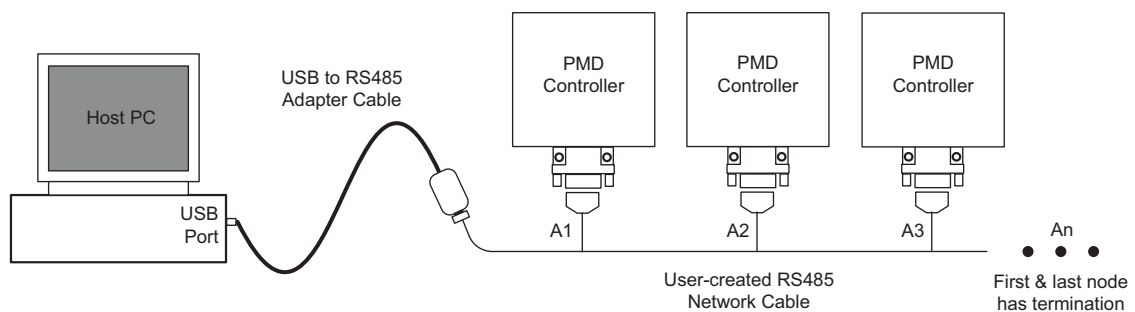
DK58113 boards support RS485 communications at connector J5. To operate with an RS485 protocol jumper JP4 must be set to the 2-3 position. Communicating by RS485 is typically less of a 'plug and play' process than communicating by RS232, and requires the user to create their own cables. For more information on the DK58113 board's RS485 interface refer to the Section 4.2.1, "RS232/485 ."

RS422 is essentially a subset of RS485 which uses a similar differential electrical interface but has only a single node connection. We will therefore focus on RS485 and show how to wire a network of RS485-connected DK58113 boards.
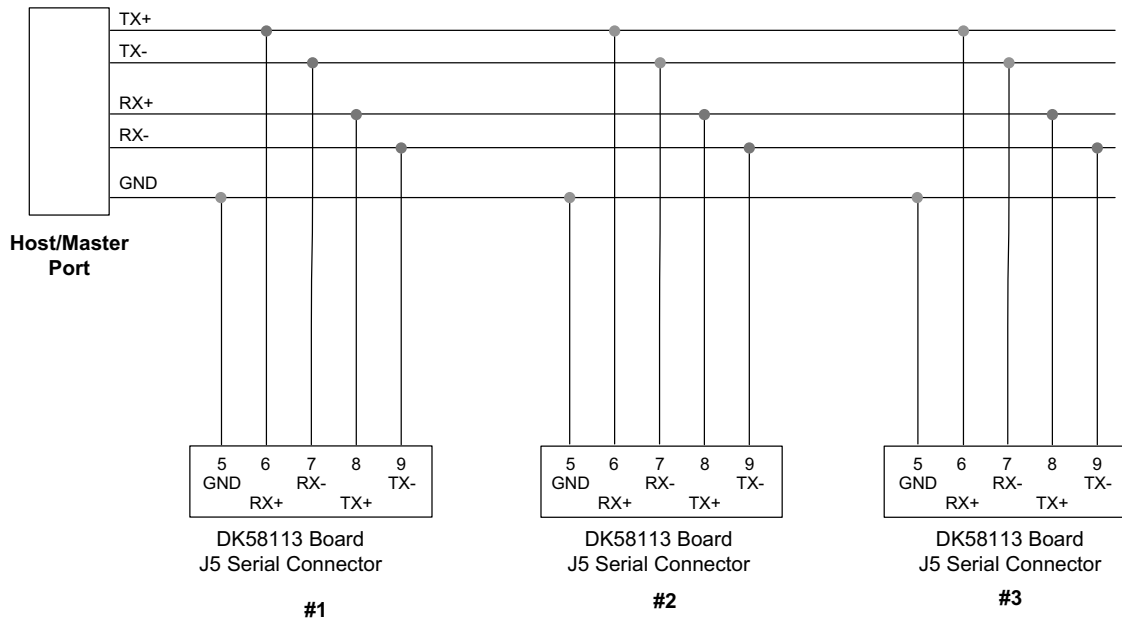
Figure 5-3:
**DK58113
Board RS485
Signal
Connection
Diagram**

RS485 is a master slave system. When connecting to a PC running Pro-Motion the PC functions as the master, and each PMD controller in the network functions as a slave. Figure 5-3 shows the wiring scheme that is used for a full duplex (four-wire) connection.

For reliable communications the last unit on both sides of the network cable should provide termination. For most systems 120 ohms of termination resistance is recommended. Additional details such as what wire type to use, wire shielding, twisting differential wire pairs, length of wire etc. will not be detailed here but are important so a reference text on RS485 networks should be consulted.

To prepare the PC to function as the RS485 master an RS422/485 USB adapter or RS485 card should be purchased and installed. Follow the product directions to install the software drivers. If properly installed and visible to Windows, Pro-Motion should recognize the RS485 port and be able to communicate with it automatically. For convenience the following table lists a vendor and P/N for such an adaptor product:

Vendor	P/N	Description
Advantech	BB-USOPTL4	Isolated high retention USB to RS422/485 converter (USB cable included)

5.1.7   RS485 NVRAM Settings

Each PMD controller unit to be installed in the RS485 network needs to be programmed with network settings. These settings can be programmed using Pro-Motion communicating to the PMD controller via the RS232 USB to DB9 cable.

To permanently store RS485 settings in the PMD controller these settings must be saved in the controller's NVRAM. For MC58113 ICs this is accomplished with Command window scripts.See Section 5.1.3, "Storing Communication Settings in NVRAM" for more information.

RS485 NVRAM settings are specified using a script with the command **SetSerialPortMode**. The complete format of this command is detailed in the *C-Motion Magellan Programming Reference*, but for convenience the table below provides the script line entries to program various common RS485 settings.

Baudrate	# Data Bits-Parity-#Stop bits	Protocol	Address	Corresponding Script Line
57,600	8-N-1	Multi-drop	0	SetSerialPortMode 0x0014
460,800	8-N-1	Multi-drop	0	SetSerialPortMode 0x0017
460,800	8-N-1	Multi-drop	1	SetSerialPortMode 0x0817
460,800	8-N-1	Multi-drop	2	SetSerialPortMode 0x1017
460,800	8-N-1	Multi-drop	3	SetSerialPortMode 0x1817
460,800	8-N-1	Multi-drop	4	SetSerialPortMode 0x2817
460,800	8-N-1	Multi-drop	5	SetSerialPortMode 0x3017

Here are the steps for storing RS485 settings into the PMD controller's NVRAM using scripts and activating these settings in the unit:

1   Compose a text script containing the command line that provides the desired RS485 settings. If not in the table above refer to the *C-Motion Magellan Programming Reference*.

2   With Pro-Motion communicating successfully to the PMD Controller via RS232, select the NVRAM button of the Pro-Motion Device Control window. Specify the file name to download, click Download!, and when the download is completed click Close

3   Disconnect existing RS232 communications using the Disconnect toolbar icon

4   Power down the PMD Controller.

5.1.8   Pro-Motion RS485 Settings

With a RS422/RS485 USB adapter or other RS485 network card installed in the PC and drivers loaded, with the DK58113 board's JP4 jumper set to the 2-3 position, and with a cable connecting the USB RS485 adapter to the DK58113 board's J5 connector installed, the instructions below will connect the host PC running Pro-Motion via RS485 to the PMD controller programmed for RS485 communications.

To set Pro-Motion RS485 parameters:

1   If not already, power up the PMD controller and click the Connect toolbar icon.

2   Select Serial from the list of options.

3   Set the PC comm port that the RS485 connection is located at and press OK

4   Enter the RS485 communication settings. The programmed baud rate should match the RS485 parameters entered in the previous section, the protocol should be programmed to multi-drop, and the address of the PMD controller you are connecting to should be entered.

5   When complete, click OK.

If RS485 communication is successful a graphical icon representing the PMD controller on the network will be loaded into the Project window. If communication is not successful, a Communications Timeout Error dialog box appears. If this happens, recheck the connections, and retry to establish communications with the PMD controller.

If after rechecking your connections communication is still not correctly established you may want to connect via RS232 and check your RS485 settings. This can be accomplished by powering down the PMD controller, setting the JP4 jumper to the 1-2 position, re-installing the USB to DB9 cable in J5, powering up the PMD controller, and selecting Connect from the top icon menu bar. Set the communication settings to the same address, baud rate, and

parity as set in Section 5.1.7, "RS485 NVRAM Settings" and set the protocol to multi-drop. Communications should re-establish and you can now check or change your RS485 settings using the instructions in Section 5.1.7, "RS485 NVRAM Settings" and then retry to connect using the RS485 cable as detailed in Section 5.1.8, "Pro-Motion RS485 Settings."

If serial communication can not be restored it is recommended that you try to connect via CAN, if available, by setting Pro-Motion to communicate at the CAN default settings. For the default communication settings for CAN refer to Section 4.2.3, "CANbus." Using the CAN connection you can try to diagnose why serial communications are not working.

5.1.9 Setting Up Multiple Units on an RS485 Network

Section 5.1.7 and Section 5.1.8 show how to program a single PMD controller and configure Pro-Motion so that correct communication are established. This is useful to verify that the RS485 network cabling and RS485 communication settings are correct.

To set up multiple PMD controllers on the RS485 network a typical sequence used is as follows:

- Program every unit that will be on the network, one by one, using the procedure in Section 5.1.7, "RS485 NVRAM Settings" however do not yet install the RS485 cables and do not execute the last step powering the PMD controller on. Each unit on the network must have a unique address and be programmed with the same baud rate.

- Install all of the programmed PMD controller units into the RS485 network, connect the RS485 network to the PC, and provide power to all of the PMD controllers on the RS485 network.

- Configure Pro-Motion to connect to each unit on the network, one-by-one, using the procedure described in Section 5.1.8, "Pro-Motion RS485 Settings." So initially one unit will be connected on the network, then a second, etc. until all units are connected

If successful, at the end of this procedure the Project window will display the entire list of units on the network, and by selecting (clicking on) a particular unit you can talk to that PMD controller directly. Once the network is installed and configured in Pro-Motion you can save this network configuration using the *File/Save Project* menu function. To recall this configuration use the *File/Open Project* function.

5.2 CAN Communications

The DK58113 board supports the CANbus 2.0 communication standard. For more information on the DK58113 board's CAN interface refer to Section 4.2.3, "CANbus."

5.2.1 CAN Hardware Setup
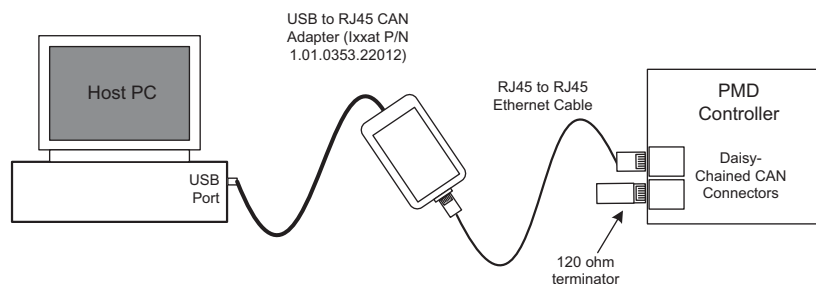
Figure 5-4: Hardware Setup for Communicating via CAN

For your PC to communicate via CAN you will typically use a USB to CAN converter which provides an RJ45 interface. The DK58113 board directly accepts the RJ45 CAN connector at either J6 or J7. Either socket can be used because each signal is daisy chained from one connector to the other.

For the most reliable communications both ends of the CAN network should provide termination. For most systems 120 ohms of termination resistance is recommended. For connection to a single DK58113 board the TRM-RJ05-02 termination device, which is included with the developer kit, can be inserted into the unused J6 or J7 CAN connector. To order more TRM-RJ45-02 devices contact your local PMD representative.

> Although not recommended for the production application, CAN network communication will often function correctly even if only one end of the network is terminated. So for bench-top checkout and application development use of only one terminator may be acceptable.

The following table provides a list of commercially available products that you may find useful for CAN communications:

Vendor	P/N	Description	Included with Machine Controller DK
Ixxat	1.01.0281.12002	USB to CAN connector, V2 IXCAN, RJ45 interface	No
CableWholesale	10x8-56101	6' Cat 6 RJ45 to RJ45 connector cable	Yes

5.2.2   CAN NVRAM Settings

Each PMD controller unit to be installed in the CAN network needs to be programmed with network settings. To permanently store CAN settings in the PMD controller these settings must be saved in the controller's NVRAM. For MC58113 ICs this is accomplished with Command window scripts.

CAN NVRAM settings are specified using a script with the command **SetCANMode**. The complete format of this command is detailed in the *C-Motion Magellan Programming Reference*, but for convenience the table below provides script line entries to program various common CAN settings.

Bitrate	NodeId	Corresponding Script Line
20,000	0	SetCANMode 0xC000
1,000,000	0	SetCANMode 0x0000
1,000,000	1	SetCANMode 0x0001
1,000,000	2	SetCANMode 0x0002
1,000,000	3	SetCANMode 0x0003

Refer to for more information about storing communication settings in the motion IC's NVRAM

Here are the steps for storing CAN settings into the PMD controller's NVRAM using scripts and activating these settings in the unit:

1   Compose a text script containing the command line that provides the desired CAN settings. If not in the table above refer to the *C-Motion Magellan Programming Reference.*

2   With Pro-Motion communicating successfully to the PMD Controller via RS232, select the NVRAM button of the Pro-Motion Device Control window. Specify the file name to download, click Download!, and when the download is completed click Close

3 Disconnect existing RS232 communications using the Disconnect toolbar icon

4 Power down the PMD Controller, wait a few seconds, and repower the PMD Controller

5.2.3 Pro-Motion CAN Settings

With a CANbus 2.0 adapter and driver installed in the PC, and with CAN cable connections in place, the instructions below will connect the host PC running Pro-Motion via CAN to the PMD controller programmed for CAN communications.

To setup Pro-Motion to communicate by CAN:

1 With no RS232 connections to the PMD controller operating, click the Connect toolbar button.

2 Select CAN, and then click OK.

3 Enter the same baud rate and Node ID as was programmed into the PMD controller previously.

4 When complete, click OK.

If CAN communication is successful, a set of graphical icons representing your PMD controller will be loaded into the Project window. If communication is not successful, after about 30 seconds, a Communications Timeout Error dialog box appears. If this happens recheck your connections and retry to establish CAN communications. If desired, you can reconnect via the RS232 connections to check or set the CAN settings as detailed in Section 5.2.2, "CAN NVRAM Settings."

5.2.4 Setting Up Multiple Units on a CAN Network

Section 5.2.2 and Section 5.2.3 show how to program a single PMD controller and configure Pro-Motion so that CAN communication is established.

To set up multiple units on the CAN network a typical sequence used is as follows:

• Program every unit that will be on the network, one by one, using the procedure in Section 5.2.2, "CAN NVRAM Settings" however do not power the PMD controller on. Each unit on the network must have a unique Node ID and be programmed with the same baud rate.

• Install all of these programmed units into the CAN network and provide power to all PMD controllers on the network.

• Configure Pro-Motion to connect to each unit on the network, one-by-one, using the procedure described in Section 5.2.3, "Pro-Motion CAN Settings." So initially one unit will be connected on the network, then a second, etc. until all units are connected.

If successful, at the end of this procedure the Project window will display the entire list of units on the network, and by selecting (clicking on) a particular unit you can talk to that PMD controller directly. Once the network is installed and configured in Pro-Motion you can save this network configuration using the *File/Save Project* menu function. To recall this configuration use the *File/Open Project* function.

5.3 SPI Communications

The DK58113 board supports SPI (Serial Peripheral Interface) communications. For more information on the DK58113 board's SPI interface refer to Section 4.2.2, "Serial Peripheral Interface."

5.3.1   SPI Hardware Setup
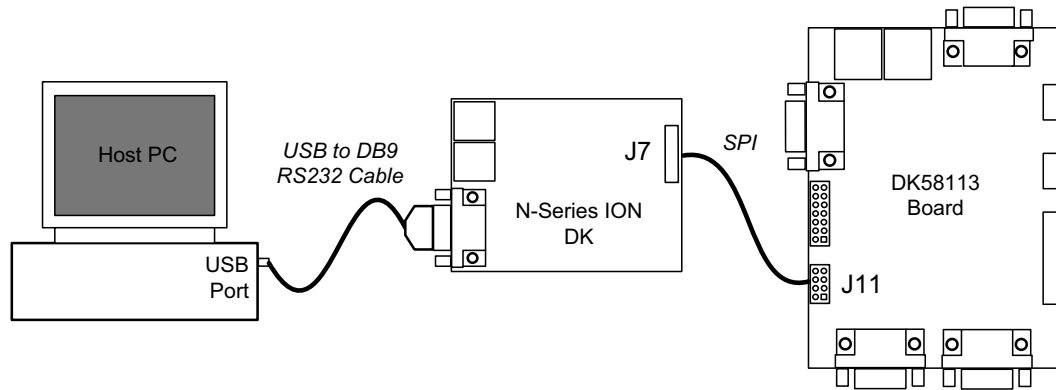
Communicating by SPI between the PC and a PMD controller may be useful during code development, or as a communications hardware reference when developing a user-designed board that will use SPI to communicate to the motion IC.

The figure above shows the recommended approach to achieving this which is a PC connected to an ION/CME N-Series ION Drive which in turn connects to a PMD controller via SPI. In this scheme the N-Series ION acts as a communications bridge and is not used to control a motor. The PC to N-Series ION connection can be Serial, CAN, or Ethernet, but for the purpose of the instructions below we will assume a serial connection.

To create this setup, in addition to the DK58113 board you will need a serial host type N-Series ION in developer kit format. There are three available serial host N-Series ION developer kits, PMD part numbers DK481S0056/02, DK481S0056/06, and DK481S0056/18 which differ only in their power output level and therefore will all work equally well.

> Communicating by SPI between the PC and a PMD controller is not recommended in the production application due to SPI's relative lack of noise immunity when connected via cables.

For additional information on N-Series ION drives and developer kits refer to *ION/CME N-Series Digital Drive User Manual* and *ION/CME N-Series ION Digital Drive Developer Kit User Manual.*

5.3.2   N-Series ION to DK 58113 DK SPI Cable Connections

To connect the N-Series ION to the DK58113 board via SPI a cable for this purpose must be assembled by the user. The table below shows the required connections between the N-Series ION and the DK58113.

N-Series ION Signal Name	N-Series ION J7 Connector Pin #	DK58113 Board Signal Name	DK58113 Board J11 Connector Pin #
ExpSPIXmt	2	HostSPIRcv	2
ExpSPIRcv	3	HostSPIXmt	1
ExpSPIClock	4	HostSPIClock	3
EXPSPICS1	5	HostSPIEnable	4
ExpSPIStatus	8	HostSPIStatus	6
GND	10	GND	7

On the N-Series ION DK side wires are connected via terminal screws, and on the DK58113 board side the connections terminate in an 8-pin double-row female header, 0.1" spacing. The assembled cable should be of shielded type and should be as short as possible to maximize communication reliability.

5.3.3   Setting Pro-Motion for Serial Communications with the N-Series ION

Here is the sequence used to set up the Pro-Motion for Serial communications with the N-Series ION.

1   With the USB to DB9 serial cable installed in the N-Series ION and power applied, click the Connect toolbar button in Pro-Motion.

2   Click Serial, select the com port that the N-Series ION DK is connected to, 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.

3   Click OK without changing any of these settings. If serial communication is correctly established, a set of object graphics for the N-Series ION you are configuring will load into the Project window.

In the above instructions we have assumed the default serial connection parameters of the N-Series ION are adequate. To change the connection parameters such as the baud rate, you will need to change the serial settings in both the N-Series ION unit and Pro-Motion. Refer to the *ION/CME N-Series Digital Drive Developer Kit User Manual* for instructions on doing this.

5.3.4   Changing N-Series ION Pin Function Settings

Next, some of the N-Series ION pin function settings will be changed thereby enabling the N-Series ION's Expansion SPI port to communicate with the DK58113 board. For more information on N-Series ION pin functions refer to the *ION/CME N-Series Digital Drive User Manual*.

1 Click the Device Control toolbar Icon and select the I/O box from the Device Control window. A dialog will appear as shown below.
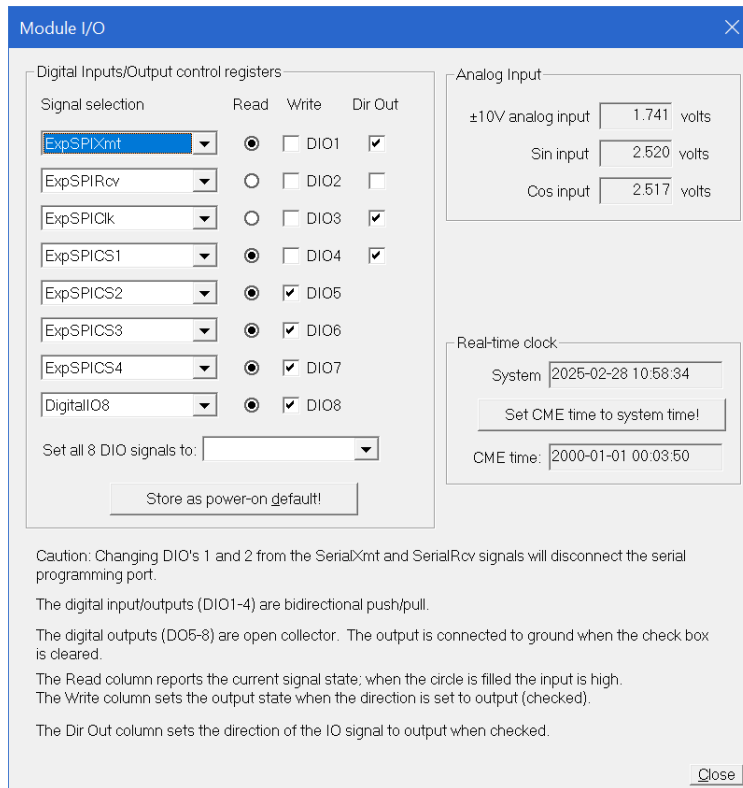
2 Select "Set all 8 DIO signals to" and set to Expansion SPI.

3 Click on "Store as power-on default!" and then click on Close.

4 Disconnect RS232 communications to the N-Series ION using the Disconnect toolbar icon.

5 Power down the N-Series ION.

# 5.3.5   Connecting to the DK58113 board by SPI

Next, Pro-Motion is connected to the N-Series ION by RS232 and then the N-Series ION's expansion SPI port is connected to the DK58113 board. In this mode where a second connection from the N-Series ION is used that connection is referred to as an attached device network.

Here are the instructions to connect to the DK58113 board via the N-Series ION's SPI attached device network:

1 Attach the USB to DB9 RS232 cable to the N-Series ION developer kit board's J12 connector and the PC's USB port, and connect the custom SPI cable to the N-Series ION DK at J7 to the DK58113 board at J11.

2 Power on both the N-Series ION and the DK58113 board.

3 Reconnect to the N-Series ION using RS232 by executing steps 1 through 5 in Section 5.3.3, "Setting Pro-Motion for Serial Communications with the N-Series ION."

4 Click the Connect toolbar icon. A dialog box will appear as shown below indicating that since you are already connected (via Serial) to the N-Series ION you have the option of connecting via an attached device network. Click Yes.
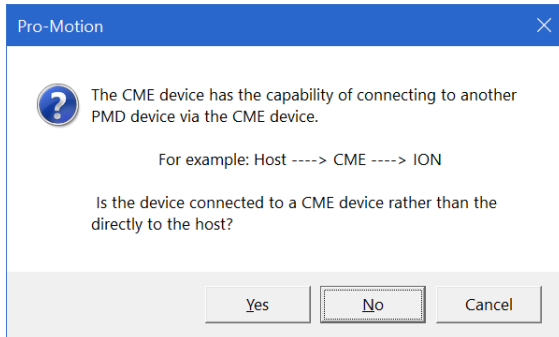

5 A dialog box will appear as shown below. Select SPI, set the protocol to Magellan, set Chip Select to 1, and specify a bit rate of 5 MHz. Then click OK.
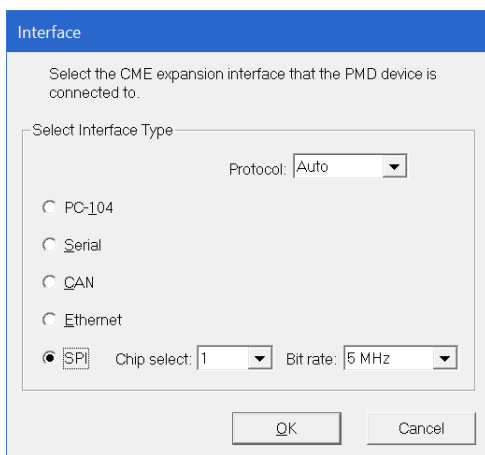
If SPI communication is successfully established a set of graphical icons representing the DK58113 board connected to the N-Series ION as an attached network will be loaded into the Project window as shown below.

If communication is not successful, after about 30 seconds, a Communications Timeout Error dialog box appears. If this happens, recheck your connections, and retry to establish SPI communications.
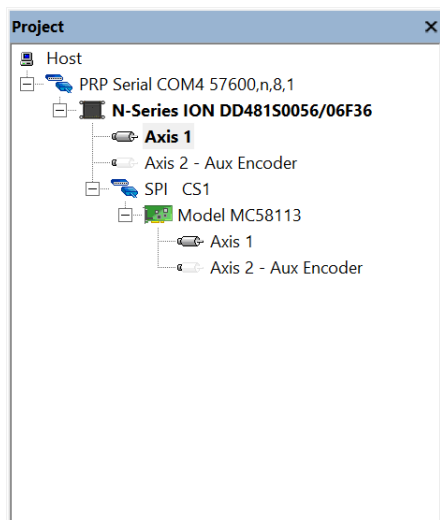
This page intentionally left blank.

6. Software Development

In This Chapter

▶ Overview of C-Motion

▶ Getting Started With C-Motion Magellan

▶ PC User Code Development

▶ Embedded User Code Development

This chapter provides a general introduction to creating software for PMD controller products. The focus will be on C-Motion, PMD's C-language libraries but PMD also provides .NET language support which enables software to be written in C# or VisualBasic.

For a complete description of PMD's software language support, tools, and design examples refer to the *C-Motion Engine Development Tools Manual*.

6.1   Overview of C-Motion

The C-language programming system used to develop user application code for PMD products is called C-Motion. C-Motion is a set of callable C-language routines that provide many features including:

- Axis virtualization

- Ability to communicate to multiple PMD motion ICs, boards, or modules

- Ability to communicate via RS232, RS485, CAN, 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 PCs, user-designed boards, or C-Motion Engines

Broadly speaking there are two different ways user application code written in C-Motion can be used to control a PMD controller; the user application program can run on a host separate from the PMD controller, or the user application program can run directly on the PMD controller in a code execution module called the C-Motion Engine. All PMD products can support user application code running on a separate host, while only PMD products with a "/CME" designation in their product name contain a C-Motion Engine.

Here is more information on these two different ways of executing the user application code.

6.1.1   Host-based Execution of User Code

When located on a host controller the user code communicates via one of the available host interfaces to the PMD controller. Depending on the PMD product being used this may be point-to-point serial, multi-drop serial, CAN, Ethernet, SPI (Serial Peripheral Interface), or PC/104. The format of these communications is one of two packet based protocols depending on the PMD product being used; PRP protocol, which is short for PMD Resource Access Protocol, or Magellan protocol. The user need not be concerned with the packet format however because these details are handled automatically when code is written in C-Motion.

For more information on the Magellan protocol refer to the *C-Motion Magellan Programming Reference*. For more information on the PRP protocol refer to the *C-Motion PRP Programming Reference*.

6.1.2 C-Motion Engine-based Execution of User Code

When executed on the PMD controller's C-Motion Engine the user code communicates internally to the resources available on the controller such as the Magellan Motion Control IC. This has speed advantages both in communicating with those resources and in real time code execution predictability. The software tools used to compile and debug C-Motion code when run on the C-Motion Engine are contained in the SDK (Software Development Kit) provided by PMD.

Executing the code directly on the C-Motion Engine allows the controller to function as a fully standalone controller. In this mode a host controller network communication link is not needed, and one or more of the PMD controller's communication ports or digital I/O ports may be used to interface to user-operated buttons or a touch screen.

Alternatively, code can be executed on the PMD controller's C-Motion Engine that processes commands which are received from a host network, thus forming an application-specific local controller within a larger system. For example for a device with a C-Motion Engine controlling a three-axis gantry a host may send high level commands which are interpreted to mean "move the gantry to location X, Y, Z". The user code executing on the C-Motion Engine parses these incoming commands and generates axis-specific motions for each of the three controlled motion axes to execute the high level host command.

6.1.3 PMD Products & User Code Execution Options

The following table shows specific types of PMD products and options for running the user application code:

PMD Product Type	Application Code Runs On	System Description
PMD Motion Control IC	Microcontroller	**User-designed board.** A microcontroller sends commands to a PMD motion control IC, both of which are located on a user-designed board. Allows standalone operation.
PMD Motion Control IC	PC, user-designed controller, or other controller	**Host-connected user-designed board.** A PC, user-designed controller, or other controller sends commands via a network connection to a PMD motion control IC which is located on a user-designed board.
PMD ION Drive or Prodigy Board	PC, user-designed controller, or other controller	**Host-connected ION drive or Prodigy board.** A PC, user-designed controller, or other controller sends commands via a network connection to a PMD ION drive or Prodigy board.
PMD ION/CME * Drive or Prodigy/CME Board	C-Motion Engine	**ION/CME drive or Prodigy/CME board.** User application code runs on the PMD controller's C-Motion Engine. Allows standalone operation.

PMD products which support a C-Motion Engine have a "/CME" in their product name, for example ION/CME N-Series Digital Drive.

6.1.4   C-Motion SDKs

There are three different C-Motion SDKs; C-Motion Magellan, C-Motion PRP, and C-Motion PRP II. All of these SDKs are available from the PMD website at http://www.pmdcorp.com/resources/software. Here is more information on each:

- **C-Motion Magellan SDK** – an SDK for creating host-based user applications for PMD products that utilize a Magellan or Juno formatted protocol.

- **C-Motion PRP SDK** – an SDK for creating host-based and downloadable C-Motion Engine-based user code for systems utilizing either a PRP (PMD Resource Access Protocol) protocol device or a Magellan/Juno protocol device. The C-Motion PRP SDK is also used in motion applications that will use the .NET (C#, VB) programming languages.

- **C-Motion PRP II SDK** – This SDK is similar to C-Motion PRP but is used with ION/CME N-Series Digital Drives. Compared to standard C-Motion PRP, C-Motion PRP II supports additional features such as multi-tasking, mailboxes, mutexes, and enhanced event management.

For reference the following table shows the packet protocol and C-Motion SDKs that can be used with each PMD product family:

Product Family	Packet Protocol	C-Motion SDKs
Magellan MC58113 Family ICs & DKs	Magellan	C-Motion Magellan, C-Motion PRP*
Magellan MC5x000 Family ICs & DKs	Magellan	C-Motion Magellan, C-Motion PRP*
Juno MC78113 Family ICs & DKs	Magellan**	C-Motion Magellan, C-Motion PRP*
ION/CME N-Series	PRP	C-Motion PRP II
ION 500 (except Ethernet)	Magellan	C-Motion Magellan, C-Motion PRP*
ION 500 with Ethernet interface	PRP	C-Motion PRP
ION/CME 500	PRP	C-Motion PRP
ION 3000	Magellan	C-Motion Magellan, C-Motion PRP*
Prodigy PC/104	Magellan	C-Motion Magellan
Prodigy/CME PC/104	PRP	C-Motion PRP
Prodigy/CME Stand-Alone	PRP	C-Motion PRP
Prodigy/CME Machine-Controller	PRP	C-Motion PRP

* With this product C-Motion PRP typically only used for .NET support, or if a mix of Magellan protocol and PRP protocol devices are attached.
** Although the Juno IC command set is somewhat different than the Magellan IC command set, the overall packet format is the same and therefore also referred to as a Magellan packet protocol.

From C-Motion Magellan to C-Motion PRP to C-Motion PRP II, each 'higher' level is a functional superset of the previous, and so when using multiple PMD products the highest required SDK should be used. For example for a PC-based application that communicates both with N-Series ION drives (which by themselves require the C-Motion PRP II SDK) and with Prodigy/CME Machine-Controllers (which by themselves require the C-Motion PRP SDK) the C-Motion PRP II SDK should be used.

Because higher SDKs are a superset of the lower level SDKs there may be situations where more than one SDK could be chosen and successfully used in the application. In subsequent sections, when this is the case, we will provide guidance to help with this choice.

6.2 Getting Started With C-Motion Magellan

C-Motion Magellan is used to develop user application code for PMD controllers that use a Magellan/Juno packet protocol. See Section 6.1.4, "C-Motion SDKs," for a list of these controller products. Magellan here refers to PMD's position control ICs such as single axis MC58113 family ICs and multi-axis MC5x000 family ICs. Juno ICs also use C-Motion Magellan however because the format and architecture of Juno commands is more or less identical to Magellan ICs.
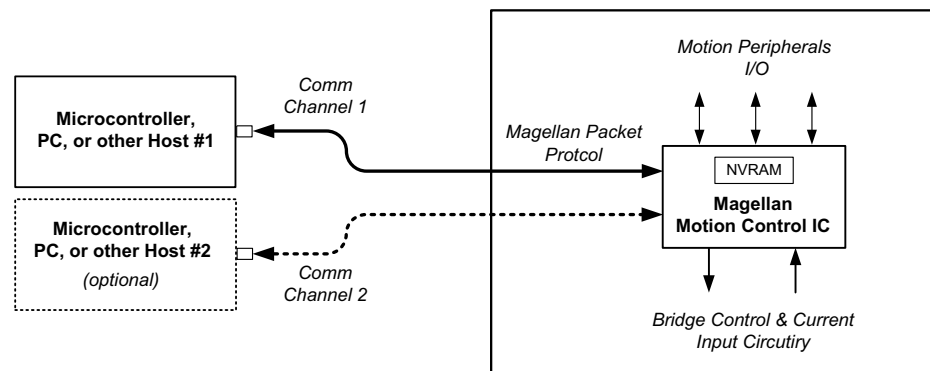
C-Motion Magellan encodes communication packets using the Magellan packet format. While the details of how Magellan packets are formatted is not important for most C-Motion users, a general understanding of the underlying architecture is helpful.

Magellan packets may be transmitted via serial, CAN, SPI (Serial Peripheral Interface) and for some products parallel-words. A basic communication to a Magellan packet format device consists of a command word and for some communications additional data words. For example for the command **SetPosition** the data words contain the 32-bit desired position value. Each packet sent to the PMD controller contains a checksum field.

After a Magellan format communication is sent a return communication is sent by the PMD controller which consists of any requested data as well as a checksum field. Commands sent to the controller that do not request data return just a checksum word.

The Magellan packet protocol is a master/slave system. The host functions as the master and initiates communication sequences which the addressed PMD controller responds to. There is one exception to this however which can occur with CAN using an 'Event' Node ID that must be different than the main communication channel Node ID.

6.2.1 Magellan Architecture Controllers

The above diagram shows connections to a generalized Magellan architecture controller. Such a controller can be an entire board product such as the DK58113 board (the developer kit board for the MC58113 IC) or the Prodigy-PC/104 board, or it can be a PMD motion control IC such as the MC58113, MC5x000, or MC78113 within a user-designed board.

In the Magellan architecture access to all available functions occurs via communication channels connected to the Magellan IC itself. This includes access to internal functions of the Magellan IC such as quadrature position tracking, trajectory generation, PWM generation, etc….

All Magellan ICs support at least two different communication interfaces, and some support as many as three. Command packet processing is simultaneously supported on each communication interface. For example a CAN communication channel can be used to command motion profiles while an RS232 connection is used to retrieve status information during the move.

This feature is particularly useful in connection with software development, as discussed later in <u>Section 6.3, "PC User Code Development,"</u> and <u>Section 6.4, "Embedded User Code Development."</u> Note that if multiple communication channels to the motion IC are being used caution should be exercised to avoid commanding motions or making other axis control changes simultaneously from both channels.

6.2.2   Accessing Magellan Functions

C-language handles are used to reference each axis supported by a particular Magellan or Juno motion control IC. This handle type is referred to as an Axis handle. For multi-axis products such as MC5x000 ICs an Axis handle will be created for each active axis, up to four. For single-axis products such as MC58113 family ICs two Axis handles are created, one for the primary axis and one for the auxiliary axis which provides just an encoder input.

Once a C-language Axis handle is obtained, that handle is used as an argument in subsequent C-Motion library calls, thereby identifying which specific axis is being commanded. This is true even if the motion system being controlled has multiple communication channels in use and multiple Magellan architecture ICs. Subsequent commands using that Axis handle will be automatically routed to the correct motion control IC via the correct communication channel.

Creating and accessing C-language Axis handles is accomplished with channel-specific C-Motion calls. For example the C-Motion call **PMDSetupAxisInterface_CAN()** returns an Axis handle to a particular axis # on a CAN network. The arguments specified with this command include a handle to an Axis and arguments related to CAN parameters such as Node ID and baud rate along with the axis # within the Magellan or Juno IC.

Additional C-Motion Magellan calls are then used to access different axes within a multi-axis motion control IC, and to add additional nodes to a network.

Refer to the *C-Motion Magellan Programming Reference* for a detailed list of C-Motion Magellan calls and reference information on Magellan packet format and syntax. For general information on Magellan Motion Control ICs and the features they provide refer to the *Magellan Motion Control IC User Guide*.

6.3   PC User Code Development

PC user code means user code that runs on the PC. This is a popular approach for controlling off-the-shelf PMD board and module products, and also for controlling user-designed boards that contain PMD motion ICs or PMD PCB-mountable drive modules.

In addition to running user code written with C-Motion libraries, running on the PC is the most popular approach when coding in C# and other .NET languages.
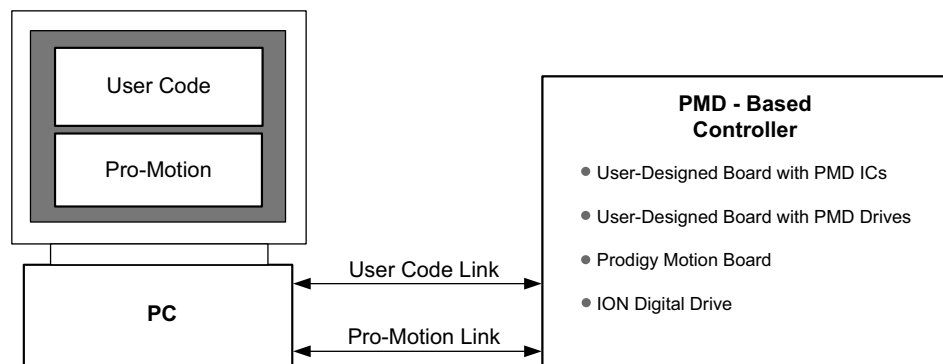
Figure 6-**2: Typical Connection Scheme for PC**-**Based User Code Development**

There are typically two separate communication links used with PC-based code development as shown in Figure 6-2:

User Code Link

When the PC runs the user application code it uses a communication link to send commands to the PMD controller. This communication channel is called the User Code link. For PRP devices, the User Code link will carry PRP-formatted packets so that the PRP device can correctly interpret them and respond accordingly. For Magellan protocol devices the User Code link will carry Magellan-formatted packets. See Section 6.1.4, "C-Motion SDKs," for a list of packet protocol types for various PMD products.

Depending on the communication channel used for the User Code link other devices may also be addressed on the same link, and these devices may or may not be a PMD controller. For example if an Ethernet TCP network is used as the User Code link the PMD controller will use just one IP Address, and other devices, PMD-based or not, may be on the same network as long as they use a different IP address.

Pro-Motion Link (optional)

The Pro-Motion link uses a separate connection to allow Pro-Motion to communicate with the PMD controller. Although optional, running Pro-Motion and utilizing a Pro-Motion link as part of the code development setup has significant benefits because Pro-Motion can be used to investigate the status of the PMD controller before, during, and after execution of user application code sequences. For more information on Pro-Motion refer to Chapter 3, *Going Further with Pro-Motion*.

> When both the user application code and Pro-Motion are running care should be taken to avoid using Pro-Motion to command motions or make changes to the PMD controller that could conflict with commands being sent by the user application code.

6.3.1   PC Link Options for DK58113 Boards

The communication link combinations that are available for different types of PMD controllers vary. The table below shows typical link combination options for User Code and Pro-Motion links when developing PC-based code to control a DK58113 board or a user-designed board with Magellan motion control ICs.

User Code Link	Pro-Motion Link
CAN	RS232
CAN	RS485
RS232	CAN
RS485	CAN

6.3.2   Choosing the SDK For PC-Based User Code Development

In most setups where the host code will run on the PC in the production application there won't be a choice of C-Motion library SDK. For example if one or more of the devices being commanded are N-Series IONs, the SDK used for the PC-based user code must be C-Motion PRP II.

The main scenario where a choice may exist is if the setup consists only of a Magellan architecture device, for example a DK58113 board (the developer kit for the MC58113 IC) or a user-designed board with PMD motion control ICs on it. In this case both C-Motion Magellan SDK and C-Motion PRP SDK are viable choices.

In this situation many developers will opt for the C-Motion PRP SDK. The reason is that if in the future a PRP device is added to the controller setup there will be no need to switch SDK types. In other words since C-Motion PRP provides a superset function of C-Motion Magellan it is inherently more 'future proof'.

The main reason for choosing C-Motion Magellan is that the size of the C-Motion source code libraries is significantly smaller. While not a major consideration if the user code runs on the PC in the production application, smaller and simpler source code libraries may be important if the user application code will eventually be ported to run on a microcontroller on a user-designed board.

For more information on C-Motion SDKs refer to Section 6.1.4, "C-Motion SDKs."

6.3.3 Code Development Process

Figure 6-3 shows a flowchart of the recommended process for developing PC-based user application code with PMD products.
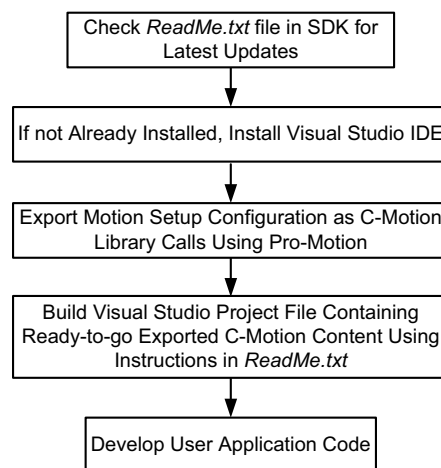
Figure 6-3: **Recommended Sequence for PC Code Development**

A first step is to review the *ReadMe.txt* file that is included at the top level directory of the SDK you are using. This file will provide up-to-date information on example Visual Studio projects, source code examples, and other resources provided by the SDK.

The next step is to install Visual Studio IDE, if you haven't already installed it. Visual Studio is a software development environment made by Microsoft Corporation. The C-Motion SDK design examples are designed to be used with Visual Studio IDE.

Next, with Pro-Motion connected to the PMD controller(s) in the motion setup and with all other configuration settings such as gain settings and safety settings in place for your motion setup, execute a configuration export to C-Motion as detailed in Section 3.10, "Configuration Export to C-Motion." The output of this operation will be one or more C-language source files that will be incorporated into your initial user application code Visual Studio project.

The instructions for the final step to building your initial user code development Visual Studio project can be found in the *ReadMe.txt* file for the SDK you are using. This step combines the C-Motion source code library files with your exported configuration source code file(s) into a ready-to-compile and run Visual Studio project.

Before executing your user code project be sure Pro-Motion is running and connected via the Pro-Motion link. As mentioned earlier Pro-Motion can be very helpful as a debugging aid by using its Command window or status screens to confirm that commands from the user application code have resulted in the expected motions or changes in the PMD controller.

As you proceed with development of your application code the following three manuals will be especially useful. For detailed information on C-Motion PRP refer to *C-Motion PRP Programming Reference*. For detailed information on

Magellan Motion Control IC commands refer to the *C-Motion Magellan Programming Reference*. For more detailed information on all aspects of developing software for PMD products refer to the *C-Motion Engine Development Tools Manual*.

6.4   Embedded User Code Development

Embedded user code means user code that runs on a microcontroller on a user-designed board that contains PMD motion control ICs such as Magellan or Juno ICs. This section provides information on developing embedded user code with PMD IC products using C-Motion libraries.

**Figure 6-4:
Typical
Connection
Scheme for
Embedded User
Code
Development**
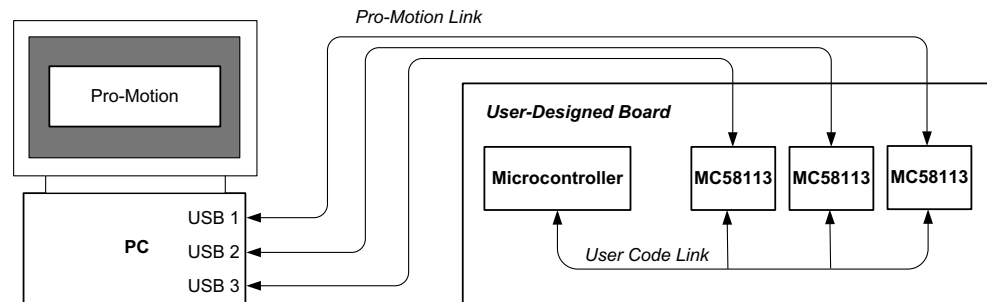
Figure 6-4 shows a typical connection scheme for embedded code development with a user-designed board.

There can be one or two communication links:

User Code Link

When a microcontroller runs the host application code it uses a communication channel to send commands to the PMD motion control IC. This channel is called the User Code link. The host microcontroller connects via one of the motion IC's available links which (depending on the PMD motion control IC used) may include point-to-point serial, multi-drop serial, CAN, SPI, or parallel-word. For direct communications with PMD motion control ICs the User Code link uses Magellan-formatted packets.

Depending on the communication type used for the User Code link other IC devices or even other off-board devices may also be addressed on the same communication link. For example if a CANbus network is used as the User Code link to connect to a PMD motion IC other PMD motion control ICs, or other non-PMD devices, may also be on the same network as long as they use a different Node ID.

Pro-Motion Link (optional)

To support a Pro-Motion link the user-designed board normally incorporates this communication channel directly into the board design. Having a Pro-Motion link during development is highly recommended because it simplifies board debugging and code development. For more information on Pro-Motion refer to Chapter 3, *Going Further with Pro-Motion*.

The Pro-Motion link uses one the PMD motion control IC's communication channels however this link must be different than the User Code link. For many embedded board designs the preferred Pro-Motion connection solution is a 3-pin UART connection from the motion IC to the PC. In this approach the only added cost to the user-designed board is a 3 pin 2mm header mating connector (example P/N Samtec MTMM-103-04-x-S-150), one for each motion control IC on the board. PMD provides a 3-pin USB to UART cable expressly designed to plug into such 3-pin connector, PMD part number Cable-USB-3P which can be ordered from your local PMD representative. Figure 6-4 above shows this type of Pro-Motion link where each motion IC is connected with its own link.

While UART serial Pro-Motion links are popular, other connection approaches and link combinations may be used which may have advantages depending on the specifics of the board design being developed. See the next section for a list of pairing options.

> When both the host user application code and Pro-Motion are running care should be taken to avoid using Pro-Motion to command motions or make changes to the PMD controller(s) that could conflict with commands being sent by the user host code.

6.4.1   Embedded Link Options for MC58113 ICs

The table below provides example connection combinations for user-designed boards that use MC58113 family ICs when both User Code and Pro-Motion links are used.

User Code Link	Pro-Motion Link
SPI	Serial (point-to-point or multi-drop)
SPI	CAN
CAN	Serial (point-to-point or multi-drop)
Serial (point-to-point or multi-drop)	CAN

6.4.2   Choosing the SDK for Embedded User Code Development

The recommended C-Motion SDK where the eventual target for code execution will be a microcontroller running on a user-designed board is C-Motion Magellan. This is the recommended SDK both for code development on the PC and code development on the microcontroller target.
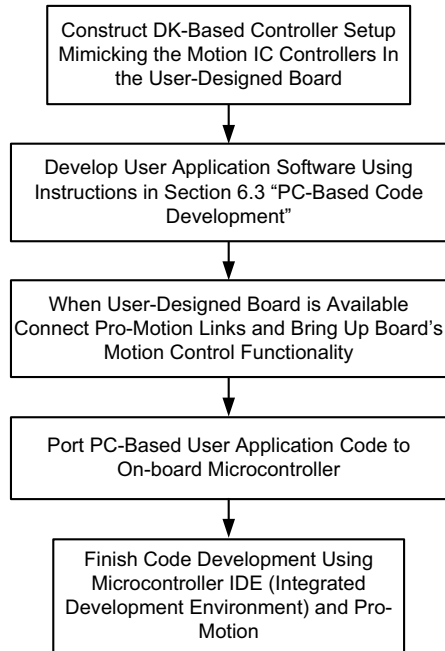
Although C-Motion PRP SDK is also a viable choice since it supports both PRP and Magellan packet communications, using C-Motion Magellan is recommended. The main reasons are that the Magellan source code libraries are smaller for the Magellan libraries than for the PRP libraries, and because the source code libraries are simpler. This will make integrating into the microcontroller environment easier, particularly if some C-Motion code sequences are executed from ISRs (Interrupt Service Routines).

For more information on C-Motion SDKs see Section 6.1.4, "C-Motion SDKs."

6.4.3   Typical Code Development Session

Figure 6-5 shows a popular development sequence for building user code that will run on the microcontroller of a user-designed board.

Construct DK-Based Controller Setup
Mimicking the Motion IC Controllers In
the User-Designed Board

Develop User Application Software Using
Instructions in Section 6.3 "PC-Based Code
Development"

When User-Designed Board is Available
Connect Pro-Motion Links and Bring Up Board's
Motion Control Functionality

Port PC-Based User Application Code to
On-board Microcontroller

Finish Code Development Using
Microcontroller IDE (Integrated
Development Environment) and Pro-
Motion

This process for developing user application code that will run on a microcontroller starts with code being developed on the PC and then later moves code execution to the microcontroller. Starting application development on the PC gives you access to resources such as Visual Studio which should make initial code development easier. It also, at least initially, avoids complications that may occur from porting the C-Motion application code to run on the microcontroller, a process that will be discussed further in Section 6.4.5, "Porting PC-Based User Code to the Microcontroller."

While initially developing code on the PC has advantages, it should be pointed out that beginning code development directly on the embedded board target may also be a good alternative approach, especially if the user has familiarity with the microcontroller IDE (Integrated Development Environment) and experience with C-Motion libraries. Readers who will use this approach can skip to Section 6.4.4, "User-Designed Board Bring-Up."

Developing and running user application code on the PC usually means that you will use developer kit boards to mimic the architecture of the user-designed board. For example if the user-designed board has three MC53113 single axis control ICs on it, you would use three DK58113 developer kit boards connected via a User Code link and a Pro-Motion link during development.

Pro-Motion Link

Pro-Motion

DK58113
Board

DK58113
Board

DK58113
Board

USB 1/Serial

USB 2/Serial

USB 3/Serial

USB 4/CAN
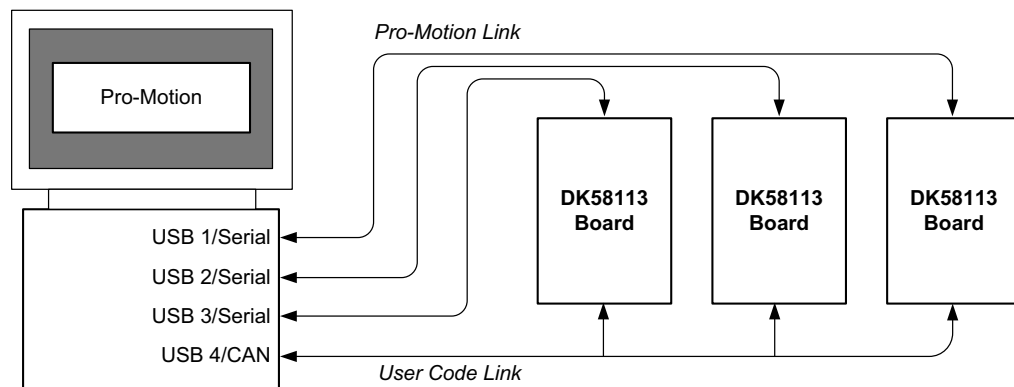
User Code Link

For most PMD motion control ICs CAN and RS485 are the available networks to communicate with multiple DK boards simultaneously. So to connect up these links the Pro-Motion link could use CAN and the User Code link could use RS485, or vice versa. Alternatively, since PCs can support multiple USB ports, USB to RS232 cables, one for each DK, can also be used for the Pro-Motion link. This is the Pro-Motion link method shown in the figure above.

For additional information on developing used code on the PC refer to Section 6.3, "PC User Code Development."

6.4.4 User-Designed Board Bring-Up

In Section 6.4.5, "Porting PC-Based User Code to the Microcontroller," we discuss fully porting the user code developed on the PC to the microcontroller on the user-designed board. Prior to that though, many developers will use Pro-Motion to assist with bring-up of the user-designed motion control board.
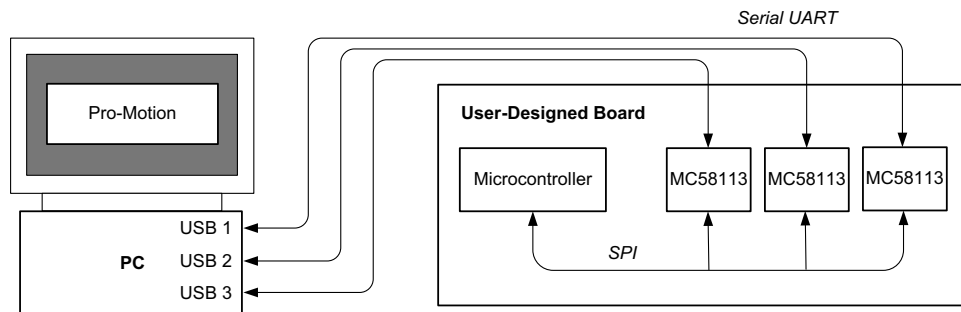
Figure 6-**7:** **Typical MC58113 User Designed Board Connections Scheme**

Figure 6-7 shows the most common connection scheme to accommodate this, using SPI (Serial Peripheral Interface) for the User Code link to each motion control IC and serial UART connections, one to each motion control IC, as the Pro-Motion link(s).

This UART connection is facilitated by the PMD accessory P/N: Cable-USB-3P which connects via a 3-pin header at the user-designed board to a USB port on the PC. Section 7.6, "3-Pin Serial UART Cable to MC58113 Family IC Connections," provides information on wiring and connecting such a 3-pin header to MC58113 family ICs.

Using these direct connections from Pro-Motion to each Magellan IC installed on the board, the motion-specific portions of the board can be exercised even before the microcontroller is up and running. Using Pro-Motion you can confirm communication with the motion IC and then check out proper functioning of each portion of motion control circuitry including the amplifier, encoder feedback, and motion peripheral signals such as home and limit switch inputs.

Once the motion control circuitry is working Pro-Motion can be used to exercise the motors driven by the user-designed board and if desired collect trace data to generate optimized motion IC control parameter settings for the user-designed board. Some users may also use these serial UART links as the User Code links, thereby allowing user application code on the PC to directly command the motion electronics on the user-designed board. Note that if the user code uses these links it is not possible to simultaneously run Pro-Motion using these same links.

6.4.5 Porting PC-Based User Code to the Microcontroller

At some point after bring-up of the user-designed board you will be ready to port the code running on the PC to run on the microcontroller. This is not an automatic process and you should check the readme.txt file included with the SDK you are using for the latest information and design examples to assist with this process.

The C-Motion SDK source code library content, as delivered, uses drivers that access the PC's COM ports. To port C-Motion to your microcontroller similar drivers for your microcontroller need to be present. PMD provides drivers

for some microprocessors and Linux in the C-Motion SDK. Check the C-Motion Magellan SDK content to determine if this is the case for your micro.

If not, you will need to write these routines yourself. To get started on this begin with a review of the C-Motion source modules that provide library calls for the communication ports you plan to use. PMDW32ser.c provides drivers for serial functions on Windows-based PCs, PMDCAN.c and PMDIXXATCAN.c provide CAN functions, and PMDNIspi.c provides SPI functions.

PMDsys.h should also be looked at because it provides execution system specific information that you may need to change. Beyond communication functions, other C-Motion functions that may need to be rewritten include time functions such as the **PMDTaskWait(ms)** C-Motion call.

**Figure 6-8:
Typical
Connection
Scheme for
Embedded User
Code
Development
Using IDE**
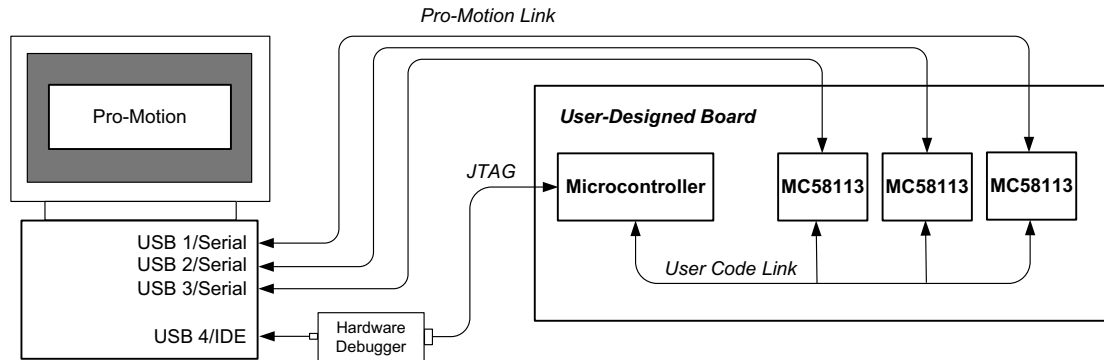
With a development setup as shown in Figure 6-8 which includes connections to the microcontroller IDE (Integrated Development Environment) that you will use to compile download, and perhaps debug executing code, you are ready to continue developing your application on the microcontroller.

Since the motion control circuitry of the board has already been checked out the next steps in the porting process are to confirm communications are functioning between the microcontroller and the on-board motion control ICs, and that the code changes to allow porting of the user code from the PC environment to the microcontroller environment are working correctly.

Once these elements have been confirmed development of the final user application code running on the user-designed board can be completed.

7. Electrical Reference

In This Chapter

▶ User-Settable Components

▶ Connectors

▶ Absolute Maximum Ratings

▶ Environmental and Electrical Ratings

▶ DK58113 On-Board Amplifier Settings Reference

▶ 3-Pin Serial UART Cable to MC58113 Family IC Connections

7.1   User-Settable Components

The following table details the available DK58113 jumper settings, which are the DK58113's only user-settable hardware components:

Jumper ID	Factory Default Setting	Setting & Description	
JP1, JP2	1-2 (on-board amplifier)	1-2	Installing jumpers at 1-2 for JP1 and JP2 configures the DK58113 for operation of the on-board amplifier.
		2-3	Installing jumpers at 2-3 for JP1 and JP2 disables the on-board amplifier, and configures the DK58113 for operation with a user-designed amplifier via the J12 Remote Amplifier Connector, or with an Atlas DK amplifier via the J10 connector.
JP4	1-2 (RS232)	1-2	Installing a jumper at 1-2 for JP4 configures the DK58113 for RS232 serial operation.
		2-3	Installing a jumper at 2-3 for JP4 configures the DK58113 for RS485 serial operation.

7.2   Connectors

**Figure 7-1:
DK58113
Board
Component
Location**
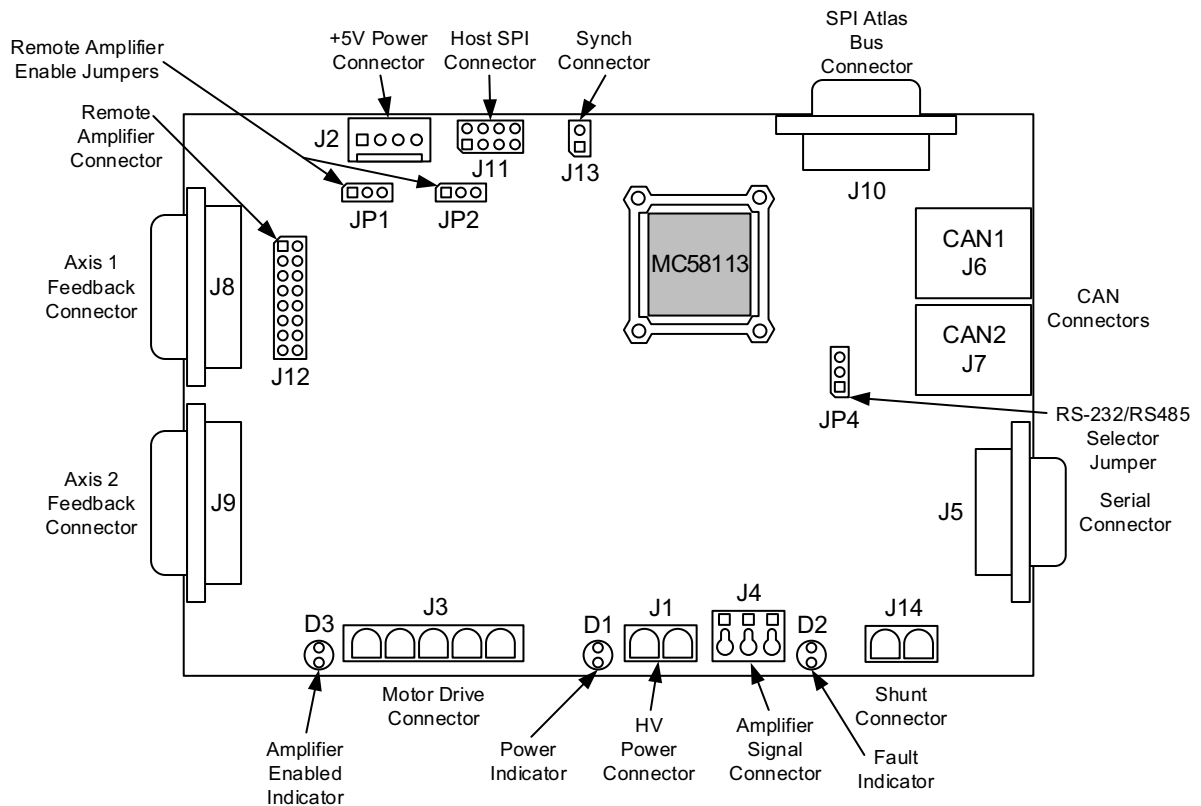
There are 14 user-accessible connectors on the DK58113 board. See Figure 7-1 for the specific locations of the connectors on the board. The connectors and their functions are outlined in the following table:

Connector Name	Connector #	Description
HV Power	J1	HV power to the board. Accepts DC supply in the range of +12 to 56 VDC
Motor Drive	J3	Motor drive signals that connect directly to the motor's coil connections
Axis 1 Feedback	J8	Primary axis feedback signals such as Quad A/B, Index, Hall A/B/C, PosLim and NegLim
Axis 2 Feedback	J9	Auxiliary axis feedback signals such as Quad A/B and Index, as well as AxisIn and AxisOut signals for the primary axis
Amplifier Signal	J4	Provides Enable input and FaultOut output signals to/from the MC58113 IC
SPI Atlas Bus	J10	Interconnect signals for SPI Atlas bus-compatible amplifiers
Remote Amplifier	J12	Provides signals to allow connection to a user-designed external amplifier
Shunt	J14	Switched HV supply voltage that connects to an external shunt resistor or other load for controlling HV overvoltage
CAN1, CAN2	J6, J7	Provides connections to a CAN2.0B host network
Serial	J5	Serial port for RS232 or RS485 host connections
Host SPI	J11	Provides host SPI (Serial Peripheral Interface) bus signals

Connector Name	Connector #	Description
Synch	J13	Provides signals that allow synchronization of the MC58113 with external controllers
+5V Power	J2	Provides logic power to the board when HV Power is not available or is not being used

7.2.1   HV Power Connector (J1)

The DK58113 board uses a dedicated 2-pin HV power connector (J1) that accepts input voltage in the range of +12 to 56 VDC. This connector is a Phoenix Contact 2-circuit pluggable terminal block connector.

Note that there is also a +5V power connector (J2) on the board. Users can power the DK58113 board with +5V supply when HV is not available. However, this will not power the on-board switching amplifier. When HV is present, the +5V supply via J2 should not be used. For connection details see

Pin	Connection	Description
		J1 - HV Power Connector
1	HV	Provides DC power to the board and on-board switching amplifier
2	GND	Ground

7.2.2   Motor Drive Connector (J3)

The Motor Drive connector (J3) provide motor output signals for use with Brushless DC, DC Brush, or step motors. This is a Phoenix Contact 5-circuit pluggable terminal block connector.

Pin	Connection	Description
		J3 - Motor Drive Connector
1	Motor A	Motor output signal A
2	Motor B	Motor output signal B
3	Motor C	Motor output signal C
4	Motor D	Motor output signal D
5	GND	Ground

7.2.2.1   Motor Connection Quick Reference

The following sections show typical motor connection names and the associated DK58113 on-board connector and pin numbers.

Brushless DC Motor Connections

Connection Name	DK58113 Pin
Motor A	J3-1
Motor B	J3-2
Motor C	J3-3
Case/Shield	J3-5

DC Brush Motor Connections

Connection Name	DK58113 Pin
Motor+	J3-1
Motor-	J3-2
Case/Shield	J3-5

Step Motor Connections

Connection Name	DK58113 Pin
Motor A+	J3-1
Motor A-	J3-2
Motor B+	J3-3
Motor B-	J3-4
Case/Shield	J3-5

7.2.3   Axis Feedback Connectors (J8, J9)

The Axis1 Feedback Connector (J8) and Axis2 Feedback Connector (J9) provide connections to various motor feedback signals. The Feedback Connectors use 15-pin high density DB connectors, which can be connected to the PMD MC-HW-05 breakout board accessories.

Axis 1 Feedback Connector

Pin	Connection	Description
		J8 - Axis1 Feedback Connector
1	QuadA1+	Axis 1 Quadrature A+ encoder input*
2	QuadA1-	Axis 1 Quadrature A- encoder input*
3	QuadB1+	Axis 1 Quadrature B+ encoder input*
4	QuadB1-	Axis 1 Quadrature B- encoder input*
5	GND	Ground
6	Index1+	Axis 1 Index+ input
7	Index1-	Axis 1 Index- input
8	Hall1A	Axis 1 Hall A input
9	Hall1B	Axis 1 Hall B input
10	Hall1C	Axis 1 Hall C input
11	Home1	Axis 1 Home input
12	PosLim1	Axis 1 Positive direction limit switch input
13	NegLim1	Axis 1 Negative direction limit switch input
14	Vcc	+5V output
15	NC	No Connect

* These signals may alternatively input pulse and direction signals to provide the axis position. To select pulse & direction signal interpretation the SetEncoderSource *command is used. In this mode Pulse signals are connected to the QuadA signal inputs and Direction signals to the QuadB inputs.*

Axis 2 Feedback Connector

Axis 2 is used to provide auxiliary encoder input for the MC58113's electronic gear mode, or with dual loop servo control mode. In addition, this connector contains the AxisIn and AxisOut signals for Axis 1 as well as a Home signal for axis 2.

Pin	Connection	Description
		J9 - Axis2 Feedback Connector
1	QuadA2+	Axis 2 Quadrature A+ encoder input*
2	QuadA2-	Axis 2 Quadrature A- encoder input*
3	QuadB2+	Axis 2 Quadrature B+ encoder input*
4	QuadB2-	Axis 2 Quadrature B- encoder input*
5	GND	Ground
6	Index2+	Axis 2 Index+ input
7	Index2-	Axis 2 Index- input
8	NC	no connect
9	NC	no connect
10	NC	no connect
11	Home2	Axis 2 Home input
12	AxisIn	Axis 1 AxisIn signal input
13	AxisOut	Axis 1 AxisOut signal output
14	Vcc	+5V output
15	GND	Ground

These signals may alternatively input pulse and direction signals to provide the axis position. To select pulse & direction signal interpretation the **SetEncoderSource** *command is used. In this mode Pulse signals are connected to the QuadA signal inputs and Direction signals to the QuadB inputs.*

Notes on Encoder Connections

Encoder inputs may be connected differentially, with two wires for QuadA, QuadB, and Index signals, or with just one wire per signal. If single-ended encoders are used, connect encoder signals to the positive encoder input only. The negative input may remain unconnected.

The following tables show this:

Encoder connections when using differential encoder input:

Signal	J8 - Axis 1 Feedback Connector	J9 - Axis 2 Feedback Connector
QuadAn+	J8-1	J9-1
QuadAn-	J8-2	J9-2
QuadBn+	J8-3	J9-3
QuadBn-	J8-4	J9-4
Indexn+	J8-6	J9-6
Indexn-	J8-7	J9-7
Vcc	J8-14	J9-14
GND	J8-5	J9-5

Encoder connections when using single-ended encoder input:

Signal	J8 - Axis 1 Feedback Connector	J9 - Axis 2 Feedback Connector
QuadAn	J8-1	J9-1
QuadBn	J8-3	J9-3
Indexn	J8-6	J9-6
Vcc	J8-14	J9-14
GND	J8-5	J9-5

7.2.4 Amplifier Signal Connector (J4)

The Amplifier Signal Connector provides an Enable input to the MC58113 IC as well as the FaultOut from the MC58113. This connector provides convenient jack screw access to these signals, making it easy to (for example) connect the GND signal to the Enable signal with a short wire.

Pin	Signal	Description
		J4 - Amplifier Signal Connector
1	Enable	Active low Enable digital input signal
2	FaultOut	Active high digital FaultOut output signal
3	GND	Ground

7.2.5 SPI Atlas Bus Connector (J10)

This connector is compatible with PMD's Atlas DK boards. See the *Atlas Digital Amplifier User Manual* for more information. This connector is a male DB-9 connector.

Pin	Signal	Description
		J4 - SPI Atlas Bus Connector
1	NC	no connect
2	NC	no connect
3	Shield	Cable Shield Connection
4	GND	Ground
5	AmpSPIRcv	AmplifierSPIRcv signal to MC58113
6	AmpSPIEnable1	~AmplifierSPIEnable signal from MC58113
7	NC	no connect
8	AmpSPIClock	AmplifierSPIClock signal from MC58113
9	AmpSPIXmt	AmplifierSPIXmt signal from MC58113

7.2.6 Remote Amplifier Connector (J12)

This connector provides signals that allow an external switching amplifier to be connected to the DK58113 board. This connector is not used if the on-board amplifier is used, or if an Atlas amplifier is used.

If a remote amplifier is used jumpers JP1 and JP2 must be set to the 2-3 position. In addition, it is not necessary to provide HV power to the board, only +5V via J2 is required.

Pin	Signal	Description
		J12 - Remote Amplifier Connector
1	AmplifierEnable1	AmplifierEnable1 output signal from MC58113
2	PWMOutputDisable	PWMOutputDisable input signal to MC58113
3	PWMHigh1A	PWMHigh1A output signal from MC58113
4	PWMLow1A	PWMLow1A output signal from MC58113
5	PWMHigh1B	PWMHigh1B/Pulse1 output signal from MC58113
6	PWMLow1B	PWMLow1B/Direction1 output signal from MC58113
7	PWMHigh1C	PWMHigh1C/AtRest output signal from MC58113
8	PWMLow1C	PWMLow1C output signal from MC58113
9	PWMHigh1D	PWMHigh1D output signal from MC58113
10	PWMLow1D	PWMLow1D output signal from MC58113
11	GND	Ground
12	AGND	Analog Ground
13	Current1A	Current1A analog input signal to MC58113
14	Current1B	Current1B analog input signal to MC58113
15	Current1C	Current1C analog input signal to MC58113
16	Current1D	Current1D analog input signal to MC58113

7.2.7   Shunt Connector (J14)

The DK58113 board uses a dedicated high current 2-pin connector for connection to a shunt resistor and diode in applications where it may be desirable to remove excess voltage from the DC bus.

This connector is a Phoenix Contact 2-circuit pluggable terminal block.

Pin	Signal	Description
		J14 - Shunt Connector
1	HV	+HV
2	Shunt	Switched connection to ground

7.2.8   CAN Connectors (J6, J7)

The DK58113 board's CAN transceivers are designed for use in applications employing the CAN serial communication physical layer in accordance with the ISO11898 standard. The transceiver provides differential transmit and differential receive capability to/from a CAN controller at speeds up to 1 Mbps.

There are two connectors, J6, and J7, providing electrically identical signals. These two connectors are designed to make it easy to connect the DK58113 board in a daisy-chain configuration. Termination at each end of the cable run is generally recommended unless cable lengths are very short and speed is slow. ISO-11898 requires 120 Ohm termination at each end of the bus. Note that it is up to the customer to verify their network topology and operating parameters.

The CANbus connector is a female RJ45 type connector.

See Section 4.2.3, "CANbus" for more information on the functionality of the CANbus port.

The pinouts for both the J6 and J7 CAN connector are as follows:

Pin	Signal	Description
		J6, J7 - CANbus Connectors
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

7.2.9   Serial Connector (J5)

The Serial Connector (J5) provides connections to an RS232 or RS485 serial port. Electrically these connectors provide access to the same signals, however they have different physical connectors and wiring. The following sections provide information for the serial connector, and provide pinouts when operated in RS232 mode or RS485 full-duplex mode.

Pin	Connection	RS232	RS485 Full Duplex
			J5 - Serial Connector
1	NC	No Connect	No Connect
2	SrlXmt	Serial transmit output	No Connect
3	SrlRcv	Serial receive input	No Connect
4	NC	No Connect	No Connect
5	GND	Ground	Ground
6	RS485Rcv+	No Connect	Positive (non-inverting) receive input
7	RS485Rcv-	No Connect	Negative (inverting) receive input
8	RS485Xmt+	No Connect	Positive (non-inverting) transmit output
9	RS485Xmt-	No Connect	Negative (inverting) transmit output

7.2.10   Host SPI Connector (J11)

This connector provides host SPI (Serial Peripheral Interface) signals when the MC58113's SPI host communication is used. This connector is an unshrouded 8-position double-row male header, .1" spacing.

Pin	Signal	Description
		J11 - Host SPI Connector
1	HostSPIXmt	Host SPI bus synchronous transmit signal from MC58113
2	HostSPIRcv	Host SPI bus synchronous receive signal to MC58113
3	HostSPIClock	Host SPI bus synchronous clock input
4	HostSPIEnable	Host SPI bus active low enable signal input
5	HostInterrupt	Host interrupt output from MC58113 can be programmed to indicate an event requiring attention from the host
6	HostSPIStatus	This signal indicates when an SPI response is available.
7	GND	Ground
8	Reset	Active low reset signal for MC58113 IC.

7.2.11 Synch Connector (J13)

This connector provides a synch signal connection which can be used to synchronize the MC58113 with external controllers including other MC58113s.

This connector is an unshrouded 2-position double-row male header, .1" spacing.

Pin	Signal	Description
		J13 - Synch Connector
1	Synch	This pin inputs or outputs a synchronization signal that can be used to synchronize the loop rates of multiple MC58113s with each other or with another external source.
2	GND	Ground

7.2.12 +5V Connector (J2)

This connector provides the DK58113 board with power to operate the on-board logic whenever HV power (J1) is not provided. Note that +5V should not be provided to the board when +HV power is provided. Connecting both power sources (+HV as well as +5V) at the same time may result in incorrect operation or damage to the DK58113 board.

The on-board switching amplifier is not operable when only +5V power is provided. The primary use of this +5V power input is when an external amplifier is used such that on-board amplification is not needed.

Pin	Signal	Description
		J2 - +5V Connector
1	+5V	+5 volts
2	+5V	+5 volts
3	GND	Ground
4	GND	Ground

7.2.13 Connector Parts Reference

The following table is supplied as a reference only.

Label	Connector Name	Connector Part Number	Connector Mate
J1	HV Power	Phoenix Contact 2-circuit pluggable terminal block p/n 1924305	Phoenix Contact 2-circuit mating terminal, 5.08mm pitch. p/n 1912401
J3	Motor Drive	Phoenix Contact 5-circuit pluggable terminal block p/n 1924334	Phoenix Contact 5-circuit mating terminal, 5.08mm pitch. p/n 1912430
J8	Axis 1 Feedback	High density D-sub 15-position, female	High density D-sub 15-position, male
J9	Axis 2 Feedback	High density D-sub 15-position, female	High density D-sub 15-position, male
J4	Amplifier Signal	Phoenix Contact 3-circuit pluggable terminal blockp/n 1985205	N/A
J10	SPI Atlas Bus	DB-9, male	DB-9, female
J12	Remote Amplifier	2 by 8 un-shrouded header, male, .1" spacing	Samtec socket, 16-position, .1" spacing. p/n: ISDM-08
J14	Shunt	Phoenix Contact 2-circuit pluggable terminal block, p/n 1924305	Phoenix Contact 2-circuit mating terminal, 5.08mm pitch. p/n 1912401
J6, J7	CAN1, CAN2	RJ45	RJ45 connector plug
J5	Serial	DB-9, female	DB-9, male
J11	Host SPI	2 by 4 unshrouded header, male, .1" spacing	Samtec socket, 8-position, .1" spacing. p/n: ISDM-04

Label	Connector Name	Connector Part Number	Connector Mate
J13	Synch	2 position unshrouded header, male, .1" spacing	Samtec socket, 2-position, .1'' spacing. p/n: ISSM-02
J2	+5V Power	Molex KK254 solid header single row, p/n 22-27-2041	Molex KK254 crimp housing, single row, 4 circuits. p/n: 22-01-2041

7.3   Absolute Maximum Ratings

HV voltage range:	0V to +60V
+5V voltage range:	-0.3V to +5.5V
Storage Temperature:	-40 to +150 C

7.4   Environmental and Electrical Ratings

Storage temperature:	-40 to +125 degrees C (-40° F to +257°F)
Operating temperature:	0 to +70 degrees C (32° F to +158°F)
HV power requirement:	+12V to + 56V operating range
Motor amplifier continuous current limit[*]:	5.0 A
Motor amplifier peak current limit:	10.0 A
Optional +5V requirement:	4.75V to 5.25V DC operating range
Digital I/O voltage range:	0V to 5V, TTL thresholds, inputs pulled up to 5V through 4.7 kOhm resistors
Digital outputs drive capacity:	Output source current: 4mA, sinking current 100mA
CAN communications:	2.0B compliant, non-isolated, 1 Mbps
Serial communications:	RS232 signaling or RS485 Full (data only)

* Current rating at 25 C ambient and with 110 CFM air flow on card. Significantly higher currents are possible with additional heat sinking. Contact your PMD representative for details.

7.5   DK58113 On-Board Amplifier Settings Reference

The DK58113 board comes with a standard MC58113 IC, and is thus not tailored for the specific amplifier and safety-related circuitry on the DK58113 board. The table below shows the correct MC58113 settings when operating the DK58113's on-board amplifier. These parameters are programmed automatically when using Pro-Motion's Axis Wizard setup sequence.

Parameter	Value & Units	Comments
Motor Output Mode	PWM High/Low	The motor output mode is set to PWM High/Low for operation with the on-board amplifier.
PWM Switching Frequency	20 kHz	This setting is motor-specific. Larger motors (some NEMA 23 and most NEMA 34) should be set for 20 kHz. Smaller motors may use 40 or 80 kHz to maximize current control accuracy and minimize heat generation.

Parameter	Value & Units	Comments
PWM Dead Time	540 nsec	For correct operation of the DK58113's on-board switching amplifier this parameter must be set to this value.
PWM Refresh Time	2,000 nSec	To ensure sufficient time to recharge the on-board amplifier's high side switches this parameter must be set to this value.
PWM Refresh Period	8 cycles	To ensure sufficient time to recharge the on-board amplifier's high side switches this parameter must be set to this value
PWM Signal Sense	Active High	For correct operation of the on-board amplifier all PWM outputs must be set to active high.
Minimum Current Read Time	2,000 nSec	To ensure sufficient minimum current read time with BLDC motors this parameter must be set to this value.
Leg Current Conversion	.733 mA/count	This value is used so that the leg current can be traced and displayed correctly in amps.
Brushless DC Motor: Foldback Continuous Current Limit	5.0 A	This value is used to ensure the on-board amplifier does not operate above the maximum safe current output.
Brushless DC Motor: Foldback Total Energy Limit	125 A^{2}sec	This value is used to ensure the on-board amplifier does not operate above the maximum safe current output.
DC Brush Motor: Foldback Continuous DC Current Limit	5.0 A	This value is used to ensure the on-board amplifier does not operate above the maximum safe current output.
DC Brush Motor: Foldback Total Energy Limit	125 A^{2}sec	This value is used to ensure the on-board amplifier does not operate above the maximum safe current output.
Step Motor: Foldback Continuous Current Limit	5.0 A	This value is used to ensure the on-board amplifier does not operate above the maximum safe current output.
Step Motor: Foldback Maximum Energy Limit	125 A^{2}sec	This value is used to ensure the on-board amplifier does not operate above the maximum safe current output.
Temperature Limit	75.0 C	This value is used to ensure the on-board amplifier does not operate above the maximum safe current output.
Bus Current Supply Conversion	.505 mA/count	This value is used so that the DC bus current supply can be traced and displayed correctly in amps.
Bus Current Supply Limit	20.0 A	This value is used to ensure safe operation of the DK58113.
Bus Current Return Limit	20.0 A	This value is used to ensure safe operation of the DK58113.
Bus Voltage Display	.966 mV/count	This value is used so that the DC bus voltage can be traced and displayed correctly in volts.
Undervoltage Limit	10.0 V	This value is used to ensure safe operation of the DK58113.
Overvoltage Limit	60.0 V	This value is used to ensure safe operation of the DK58113.

7.6   3-Pin Serial UART Cable to MC58113 Family IC Connections

PMD produces a 3-pin cable (p/n: Cable-USB-3P) which interfaces to UART level asynchronous serial signals. One end of this cable connects to a PC USB port, the other end connects to a male 3-pin 2 mm header. The most common use of this cable/header combination is to enable direct communications between a PC and a motion control IC - typically for board bring-up or for debugging.

The table below shows how the 3-pin header should be wired to connect to the serial port of MC58113 family ICs:

Cable-USB-3P Pin	MC58113 Signal Name	MC58113 Family IC Pin
1	SrlXmt	43
2	SrlRcv	50
3	GND	47

The illustration below shows the 3-pin end of this cable. To insure that the cable is installed correctly the user-designed board's silk screen should indicate the pin #1 location of the header. On the cable this pin is marked with a white dot.
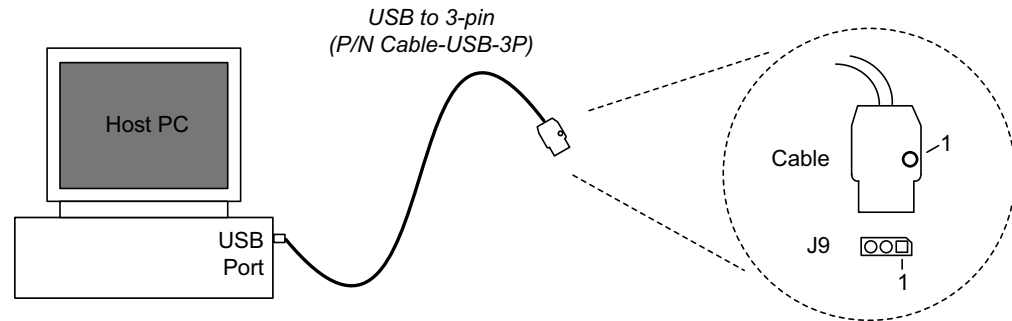
**Figure 7-2:
Cable-USB-3P
Connector**

A.DK58113 Board Schematic

Figure A-1: DK58113 Board Schematic, Magellan IC

Figure A-2: DK58113 Board Schematic, Communication Ports

RS232/RS485 TRANSCEIVER & CONNECTOR

CAN TRANSCEIVER & CONNECTORS

HOST SPI BUS DRIVER & CONNECTOR
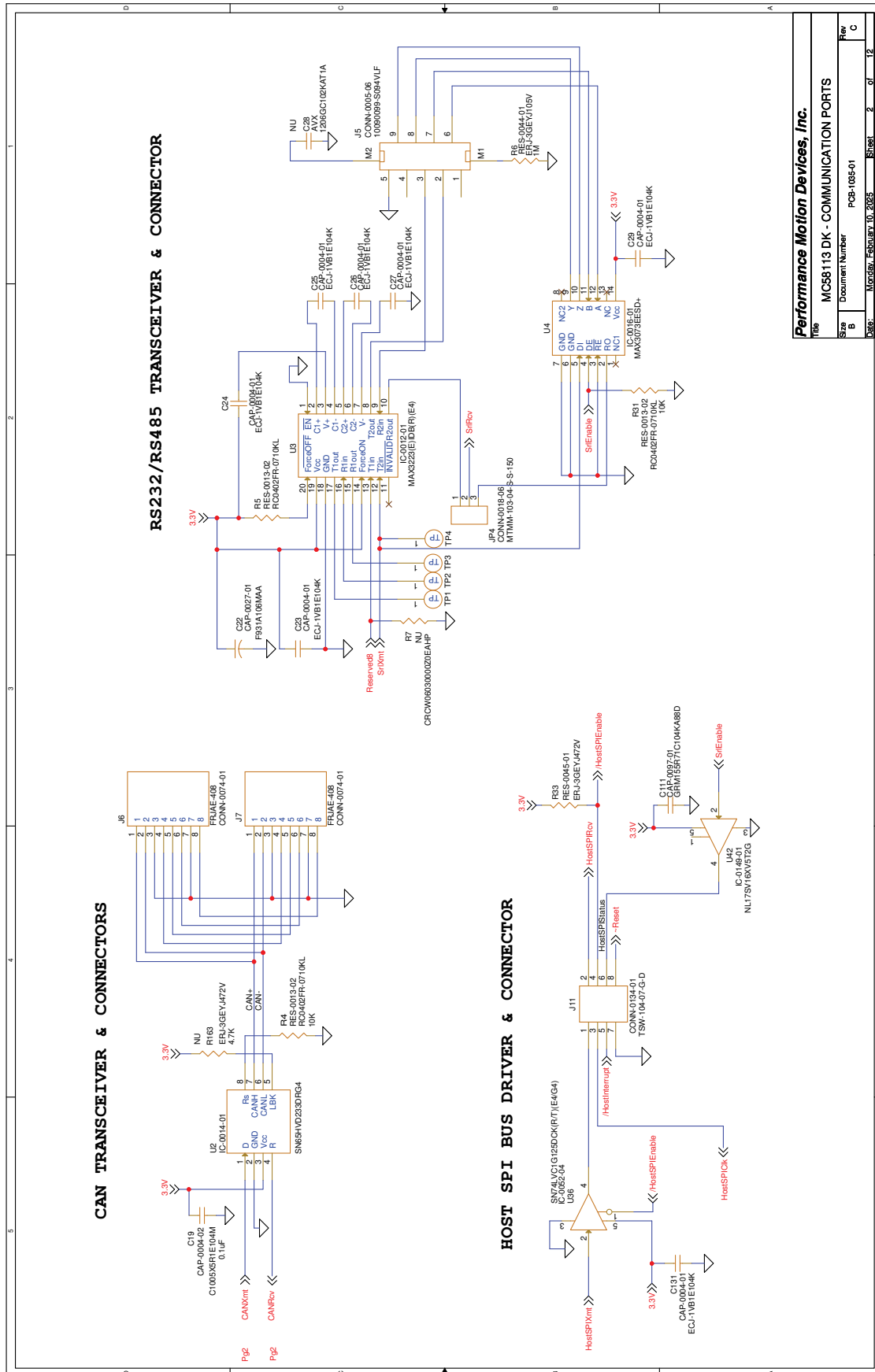
Performance Motion Devices, Inc.

Title: MC58113 DK - COMMUNICATION PORTS

Size B   Document Number PCB-1035-01   Rev C

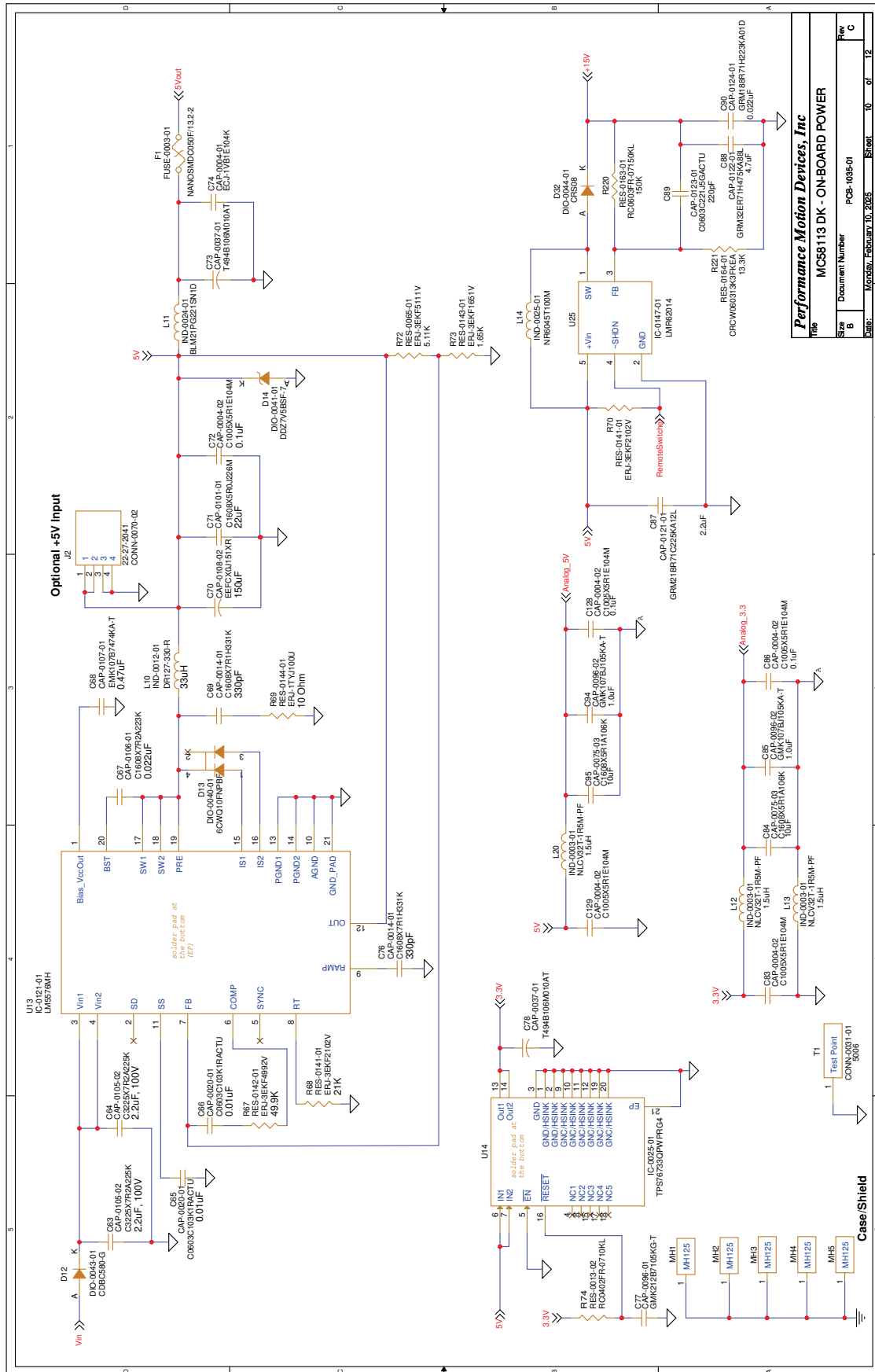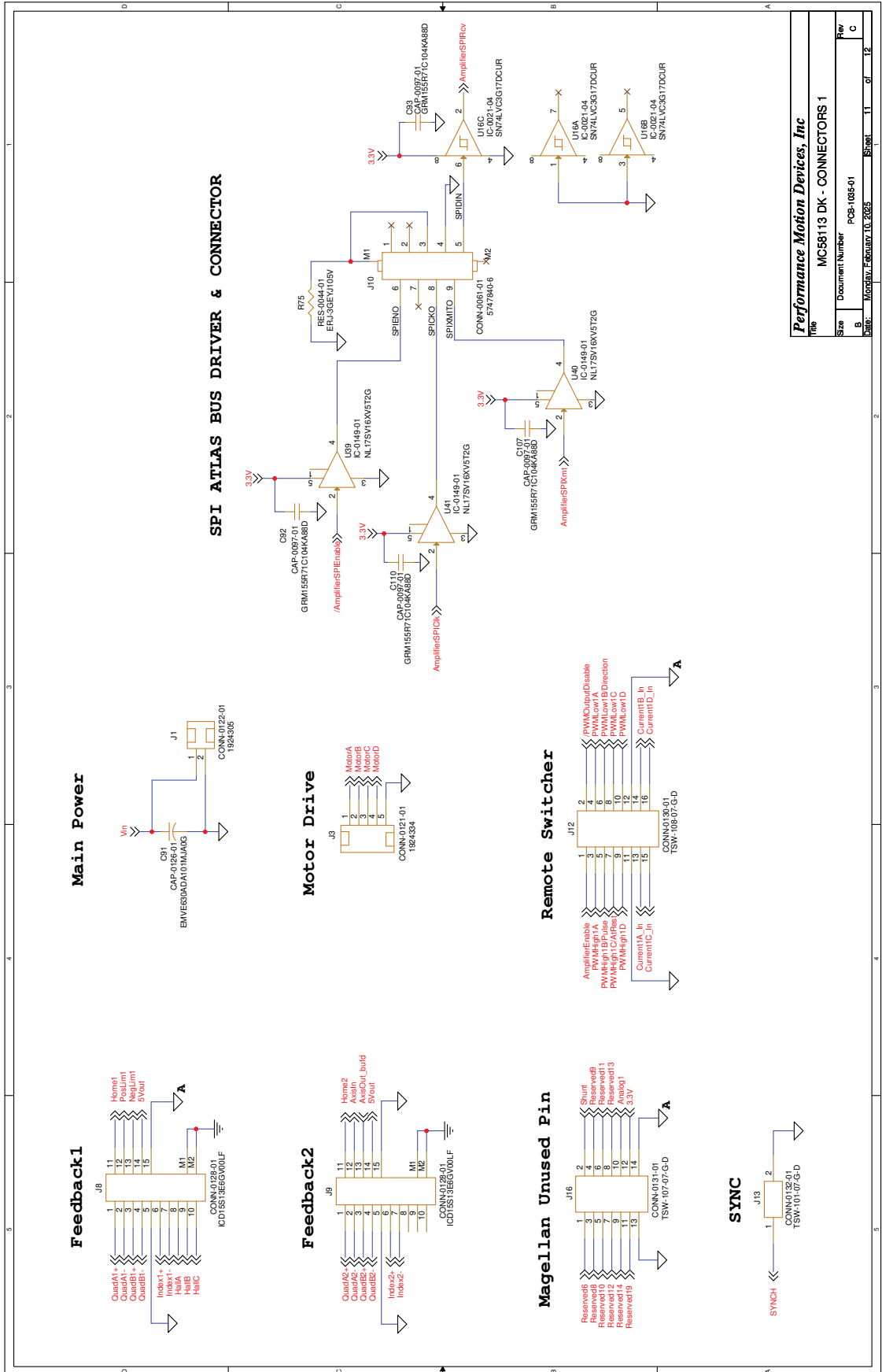Date: Monday, February 10, 2025   Sheet 2 of 12

**Figure A-3:
DK58113
Board
Schematic,
Encoder
Feedback**

DK58113 Developer Kit User Manual
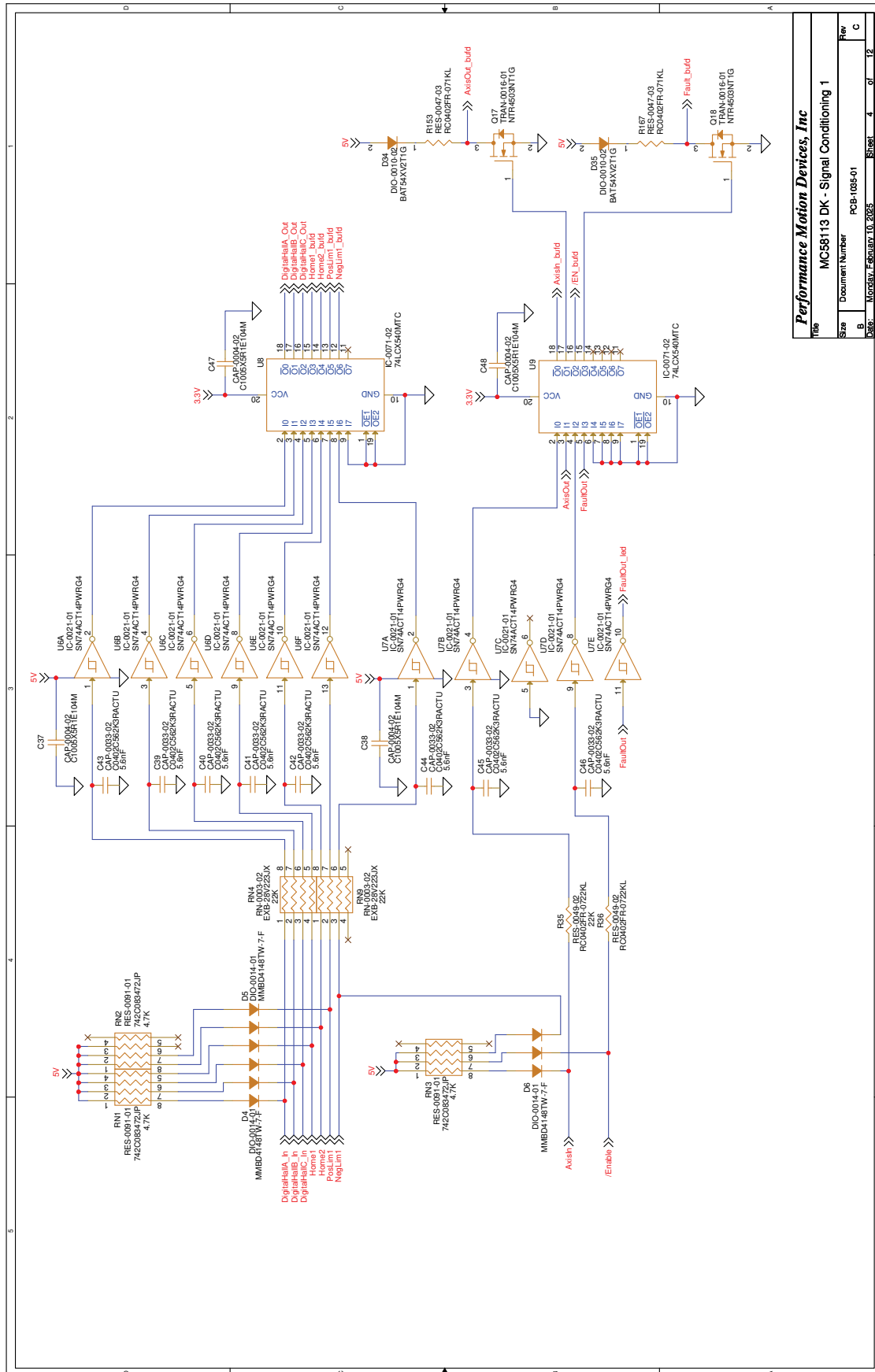
Figure A-4:
DK58113
Board
Schematic,
Signal
Conditioning 1

Figure A-5: DK58113 Board Schematic, Signal Conditioning 2

DK58113 Developer Kit User Manual

Figure A-6:
DK58113
Board
Schematic,
Phase A/B
Power Train

Figure A-7: DK58113 Board Schematic, Phase C/D Power Train
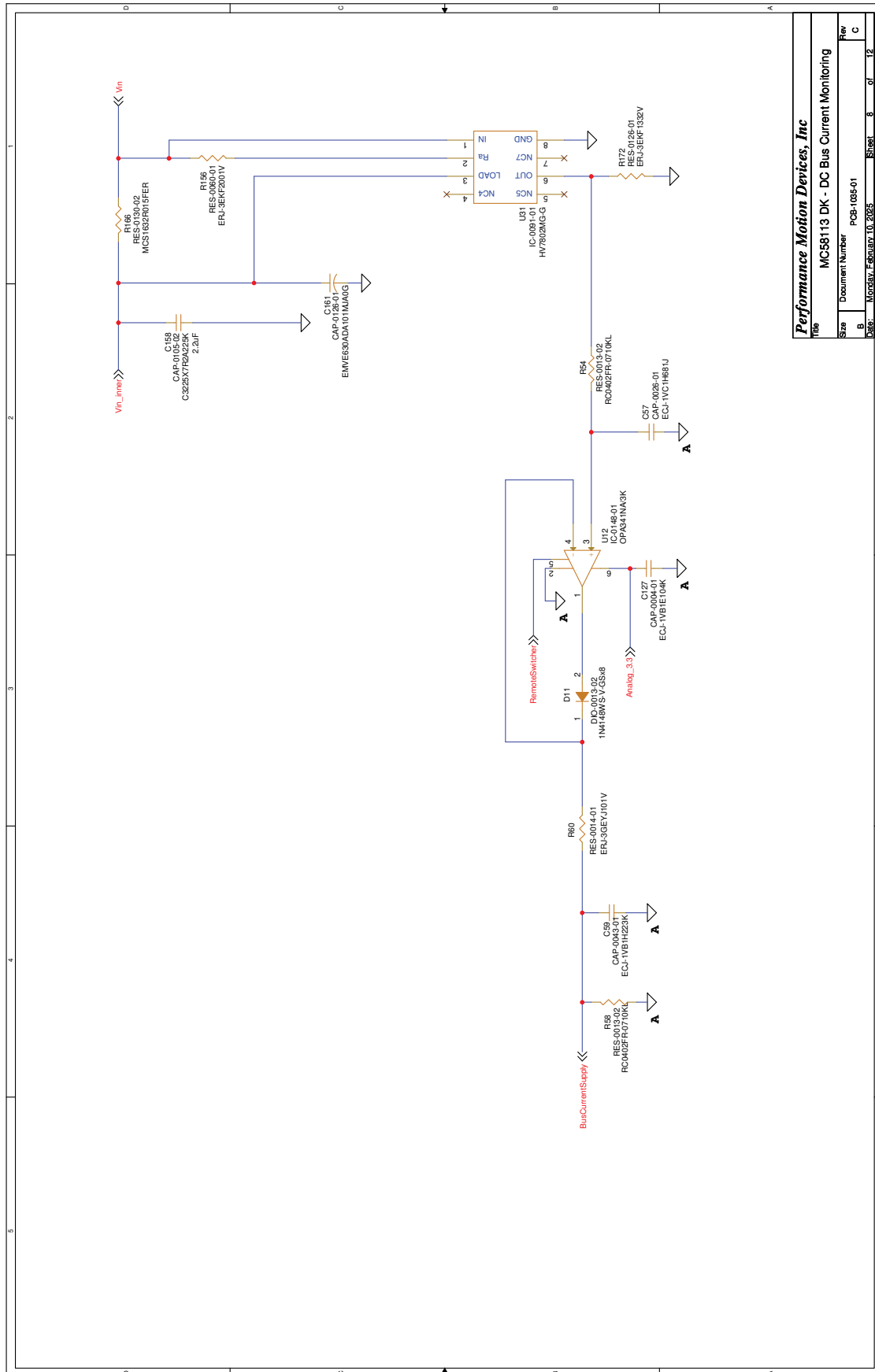
Figure A-8: DK58113 Board Schematic, DC Bus Current Monitoring

**Figure A-9:
DK58113
Board
Schematic,
Voltage
Monitoring**

Temperature monitoring

HV Voltage Sensing

Figure A-10: DK58113 Board Schematic, On-Board Power

**Figure A-11:
DK58113
Board
Schematic,
Connectors 1**

SPI ATLAS BUS DRIVER & CONNECTOR

Main Power

Motor Drive

Remote Switcher

Feedback1

Feedback2

Magellan Unused Pin

SYNC

Performance Motion Devices, Inc
Title: MC58113 DK - CONNECTORS 1
Size B Document Number PCB-1035-01 Rev C
Date: Monday, February 10, 2025 Sheet 11 of 12
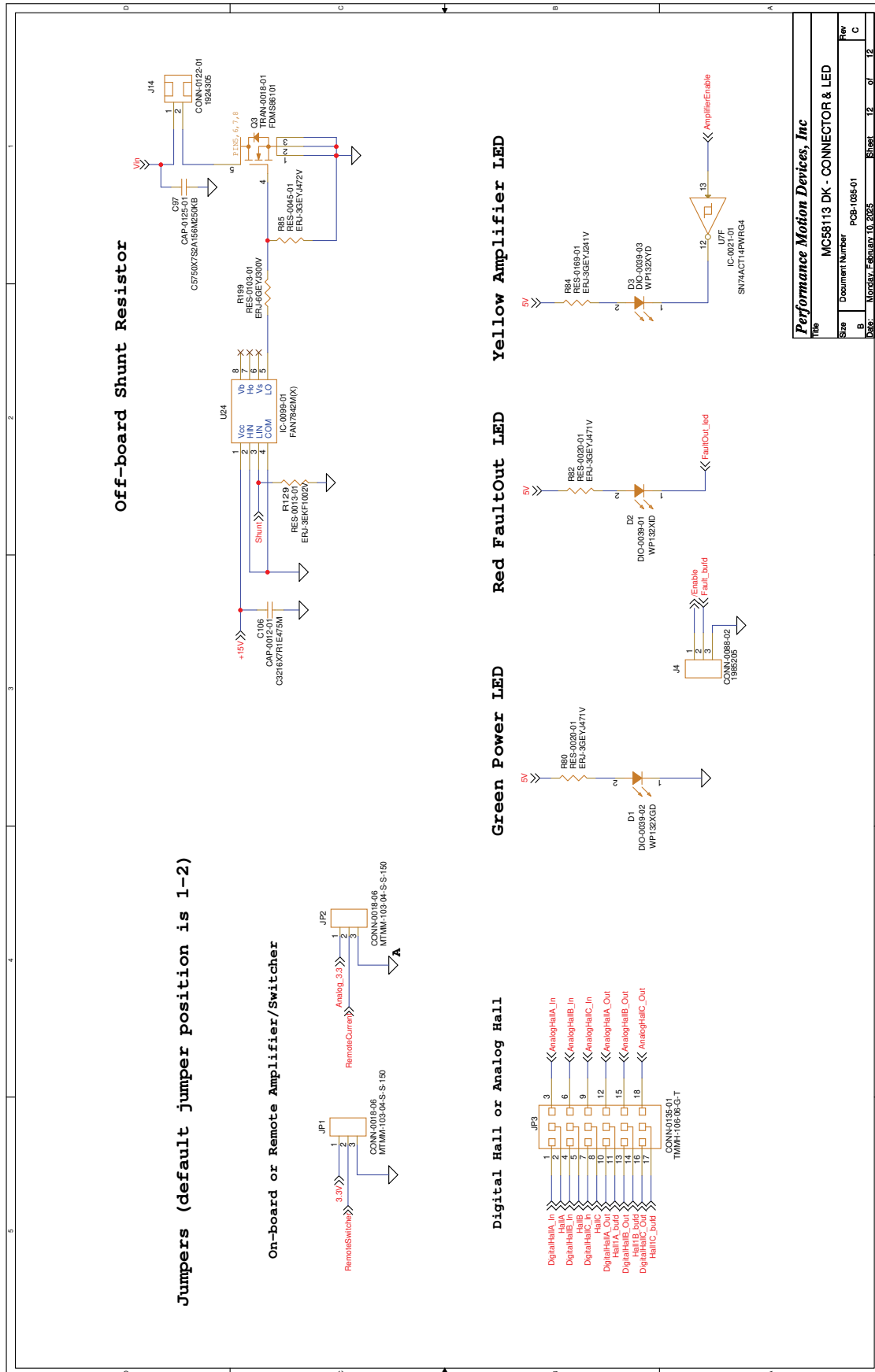
Figure A-12: DK58113 Board Schematic, Connector & LED

Off-board Shunt Resistor

Yellow Amplifier LED

Red FaultOut LED

Green Power LED

Jumpers (default jumper position is 1-2)

On-board or Remote Amplifier/Switcher

Digital Hall or Analog Hall

This page intentionally left blank.

B. Temperature Conversion Tables

In This Appendix

▶   Thermistor Function and Table Verification

Pro-Motion has the ability to input an ASCII text file containing a table allowing it to convert ADC (Analog to Digital Converter) readings from the MC58113's analog Temperature signal input (pin 31) into degrees C.

When the DK58113 board is initialized using the Pro-Motion Axis Wizard a temperature conversion file is automatically loaded so that readings from the temperature sensor on the DK58113 board are displayed correctly in degrees C. Pro-Motion can load other conversion files as long as their content follows the correct format. To load a conversion table manually use the Drive Safety button in Pro-Motion's Axis Control window and click the Drive Signal Scaling button to enter the file name.

The content of the conversion file, an example of which is shown below, consists of a series of entries with two numbers per line. The first is a floating point number with units of degrees C, and the second is an integer representing the corresponding ADC value from 0 to 32,767. The first number is separated from the second by a <Tab> character, and each line is terminated by a <NEWLINE> character.

```
Example Temp Table.txt - Notepad
File  Edit  Format  View  Help

-40.0    31417
0.0      24328
10.0     21276
20.0     18010
21.0     17683
22.0     17356
23.0     17030
24.0     16706
25.0     16384
26.0     16063
27.0     15744
28.0     15428
29.0     15114
30.0     14803
31.0     14495
32.0     14190
33.0     13888
34.0     13589
35.0     13295

Ln 20, Col 1  100%    Windows (CRLF)    UTF-8
```
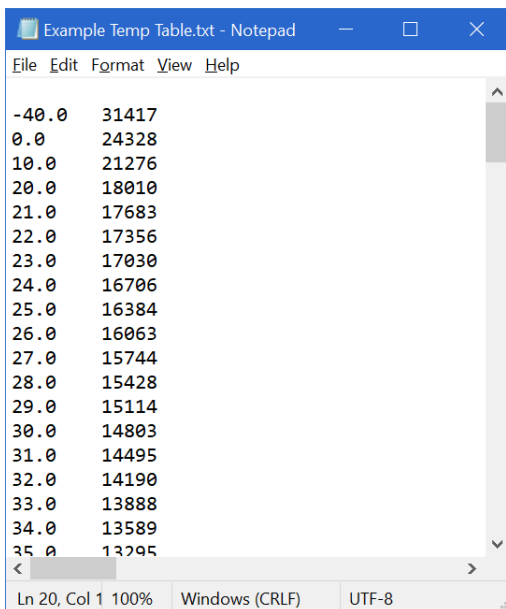
Degree entries do not need to be exactly one degree apart or even at equal temperature intervals. The only requirement is that the temperature entries be monotonic and increasing (the first entry should be the lowest temperature value and each entry after should be a higher temperature than the value immediately preceding it). Pro-Motion parses the table and linearly interpolates the values from the ADC to convert to degrees C.

Both 'increasing resistance with increasing temperature' and 'decreasing resistance with increasing temperature' thermistors can be handled by Pro-Motion. Either way the table should just list the ADC value associated with a series of increasing temperature values.

A typical circuit for processing resistance values in the thermistor into a voltage at the MC58113's Temperature signal input is shown below. This signal input expects a voltage in the range of 0V to 3.3V (unless otherwise set via the AnalogRefHigh and AnalogRefLow signals) representing the sensed temperature, with corresponding ADC value table entries in the range of 0 to 32,767.

Figure B-1: Typical Thermistor Processing Circuit
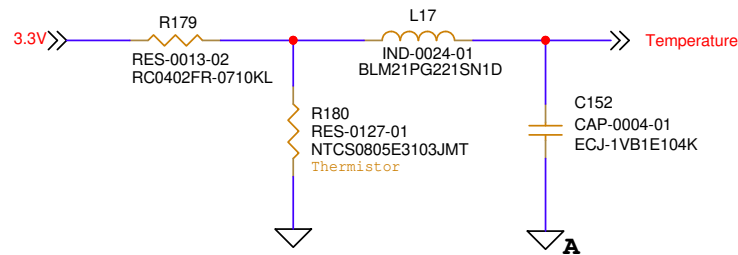
If the thermistor's data sheet shows the correspondence of temperature to resistance it is generally straightforward to construct a conversion table. For this circuit, which uses a 10 Kohm voltage divider, with a thermistor resistance of R at a particular temperature the ADC value entry TE is expressed by:

$TE = 32,767 * R/(10,000 + R)$

B.1   Thermistor Function and Table Verification

Confirming that a newly developed thermistor circuit and conversion table are working correctly is highly recommended. Accurate temperature readings are vital for the correct functioning of the MC58113's over temperature detect safety feature.

There may be several ways to do this, but in the description below a calibrated oven with a settable temperature, an operating version of the thermistor signal processing circuit connected to the MC58113, and Pro-Motion is used.

The circuitry should be powered on and located inside the oven, and Pro-Motion should be in communication with the MC58113 IC using one of the host communication channels, usually RS232. The Pro-Motion Scope window should be setup to trace two variables at the same time:

- the converted drive temperature (specify the *Temperature* variable in the Drive data category)

- the raw temperature signal input (specify the *Temperature (raw)* variable in the Analog Inputs (raw) data category

Select a relatively slow capture rate with a Trace Period of perhaps 20,000 cycles, and select a trace length of perhaps 1,000. Set the oven to the initial temperature and when it stabilizes select "Start Trace" to begin the trace. With a trace period of 20,000 cycles one data point per second will be collected. With the trace still running command the oven to the next temperature (perhaps 5 degC higher) and when the temperature stabilizes repeat for increasing temperatures until the highest desired temperature is reached. To stop the trace (if it hasn't stopped already) select "Stop Trace". Once the trace is completed you may find it useful to store the collected trace data in a spreadsheet. This can be done using the Export Trace function of Pro-Motion's File Menu item.

The advantage to simultaneously capturing both the converted temperature readings and the corresponding raw analog values is that it may help you diagnose any problems with the circuit or table if the converted temperature readings are incorrect.

An alternative, simpler approach is to just view the Pro-Motion Status window which continuously displays the converted temperature reading from the thermistor. You can visually confirm that the oven temperature and reported temperature in the MC58113 Status Window are acceptably close over the desired range of temperature readings.

This page intentionally left blank.

Index

power, applying 22
product summary 10
PWM Power Stage 66

R
R-C lowpass filter bandwidth
 /Enable and FaultOut signals 84
 Hall inputs 81
 limit and home inputs 81
RS232/485 connector communication ports 64

S
safety interlocks 84
signals
 /Enable 84
 AxisIn and AxisOut 82
 FaultOut 84
 single-ended 80
sinusoidal commutation 81
SPI bus 71
step motors, power stage configuration 66
synch I/O connectors 85

T
temperature sensors 71

U
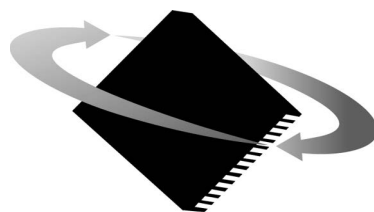undervoltage, condition and threshold 73
user-settable components 113

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

Visit our website at https://www.pmdcorp.com

P M D

Performance Motion Devices
80 Central Street
Boxborough, MA 01719