



Quick-Spin Motor Controller User Manual.

All American

MCUexperts@AllAmerican.com



© 2007 All American. All Rights Reserved.

ALL AMERICAN LEGAL NOTICE.

Sample Code and Software:

This software is provided for reference purposes only. It is provided "AS IS" without warranty or condition of any kind, either express or implied. Without limitation, All American makes no warranty with respect to merchantability, fitness for any particular purpose, lack of viruses, or accuracy or completeness of responses or results. All American also makes no warranty of title or against the infringement of any intellectual property rights of others.

Any use of this software is at your sole risk. You are responsible for ensuring the applicability of this software with your own systems. All American has no responsibility for any damage, liability or loss resulting from your use of this software. Any use by you of this software may only be with products purchased from All American. This software is not intended nor authorized for life support, space, or other mission critical applications.

This software may contain copyrighted materials, trade secrets and other All American proprietary information and is subject to copyright laws, trade secret laws and other intellectual property laws. You agree not to disseminate, decompile or reverse engineer this software.

Application Notes and Written Materials:

These materials are provided for reference purposes only. They are provided "AS IS" without warranty or condition of any kind, either express or implied. Without limitation, All American makes no warranty with respect to merchantability or fitness for any particular purposes or against the infringement of any intellectual property rights of others. This information may include technical inaccuracies or typographic errors. All American assumes no responsibility for any errors that may be in these materials, and has no obligation to update these materials. All information contained in these materials is subject to change without notice from All American.

It is your responsibility, when using any of the information contained in these materials, to evaluate such information as part of a total system. All American shall have no responsibility for any damage, liability or other loss resulting from the information contained herein. All American reference materials, including but not limited to application notes, code and software, reference designs and white papers, are not intended or authorized for life support, space, or other mission critical applications.

These materials copyright © 2007 by All American. All rights reserved. No part of these materials may be reproduced, by any means, in whole or in part, without the express written consent of All American.

WARNING!!!



WARNING!!! THE QUICK-SPIN BOARD CONSISTS OF BOTH ISOLATED AND NON-ISOLATED CIRCUITS. NON-ISOLATED CIRCUIT GROUND IS CONNECTED DIRECTLY TO ONE SIDE OF THE POWER SUPPLY INPUT. **HAZARDOUS VOLTAGES IN EXCESS OF 300 VOLTS ARE PRESENT** WHEN QUICK-SPIN IS POWERED FROM THE AC LINE. ONLY QUALIFIED PERSONS SHOULD OPERATE THE QUICK-SPIN BOARD. **FAILURE TO OBSERVE PROPER SAFETY MEASURES COULD RESULT IN SEVERE ELECTRICAL SHOCK, OR EVEN DEATH.**

WARNING!!!

WARNING!!!

WARNING!!!

WARNING!!!

WARNING!!!

WARNING!!!

Acknowledgements

This project and its accompanying workshop is produced by All American in conjunction with the following companies. Many thanks to the individuals that worked with us.

Renesas Technology of America

www.renesas.com

Fairchild Semiconductor

www.fairchildsemi.com

Contract Manufacturing:

PCB-Solutions

Layton, UT.

www.pcb-solutions.net

TABLE OF CONTENTS

	Legal.	ii
	Safety.	iii
	Acknowledgments.	iv
1.	Introduction.	7
1.1	Features.	7
1.2	Block Diagram.	7
1.4	Technical Support.	8
2.	Getting Started.	8
2.1	Kit Contents.	8
2.2	System Requirements.	9
2.3	Quick Start.	9
3.	Hardware.	9
3.1	Main Board.	9
3.1.1	Power Supply.	9
3.1.2	Motor Drive.	10
3.1.3	MCU.	10
3.1.4	Isolated Serial Port.	10
3.1.5	Temperature Sensors.	11
3.1.6	MD100 (Radiotronix module option.)	11
3.1.7	LEDs.	11
3.1.8	Connectors.	11
3.1.9	Jumpers and Options.	14
3.2	User I/O Board.	15
3.2.1	E8 Debug Interface.	15
3.2.2	Debug Isolation Bridge.	15
3.2.3	MCU.	16
3.2.4	LEDs.	16
3.2.5	Buttons.	17
3.2.6	Pot.	17
3.2.7	Connectors.	17
3.2.8	Jumpers and Options.	19
4.	Embedded Software.	20
4.1	Main Board R8C Files.	20

4.2	User I/O Board H8 Files.	20
5.	Communications Software Specification.	20
5.1	Overview.	21
5.2	Communication Protocols.	21
5.2.1	Communication medium.	21
5.2.2	Communication Protocols Overview.	21
5.2.3	Unused ASCII Characters.	22
5.2.4	Protocol Commands at a Glance.	23
5.3	Protocol A Command Set.	23
5.3.1	Protocol A Command Details.	24
5.4	Protocol B Details.	35
5.4.1	Protocol B – Main Menu.	35
5.4.2	Protocol B – Motor Parameters Submenu.	40
5.4.3	Protocol B – Speed Gain Parameters Submenu.	43
5.4.4	Protocol B – Current Gain Parameters Submenu.	44
5.3.	System Variables.	45
5.3.1	Values set by the GUI or Text Menu system.	45
5.3.2	Values created by the MCU sent to the GUI or Text Menu system.	46
6.	GUI Interface.	46
6.1	GUI Compatibility.	46
6.2	GUI Structure.	46
6.3	GUI Overview.	48
6.4	GUI Implementation in Windows 2000/XP.	48
6.4.1	GUI Configuration File.	49
6.4.2	Splash Screen.	49
6.4.1	System Startup Dialog Box.	50
6.4.2	Main Settings Tab.	52
6.4.3	Motor Parameters Tab.	54
6.4.4	PID Parameters Tab.	56
6.4.5	Set Points Tab.	57
6.4.6	PWM Information Tab.	57
6.4.7	System Settings Tab.	58
6.4.8	About Quick-Spin Tab.	59
6.5	Non Windows GUI Implementation.	60
7.	Windows GUI Code.	60
7.1	Introduction.	60
7.2	Non Windows, C++ Motor Control Object.	61
7.3	Files.	61
8.	Troubleshooting.	62

1. Introduction.

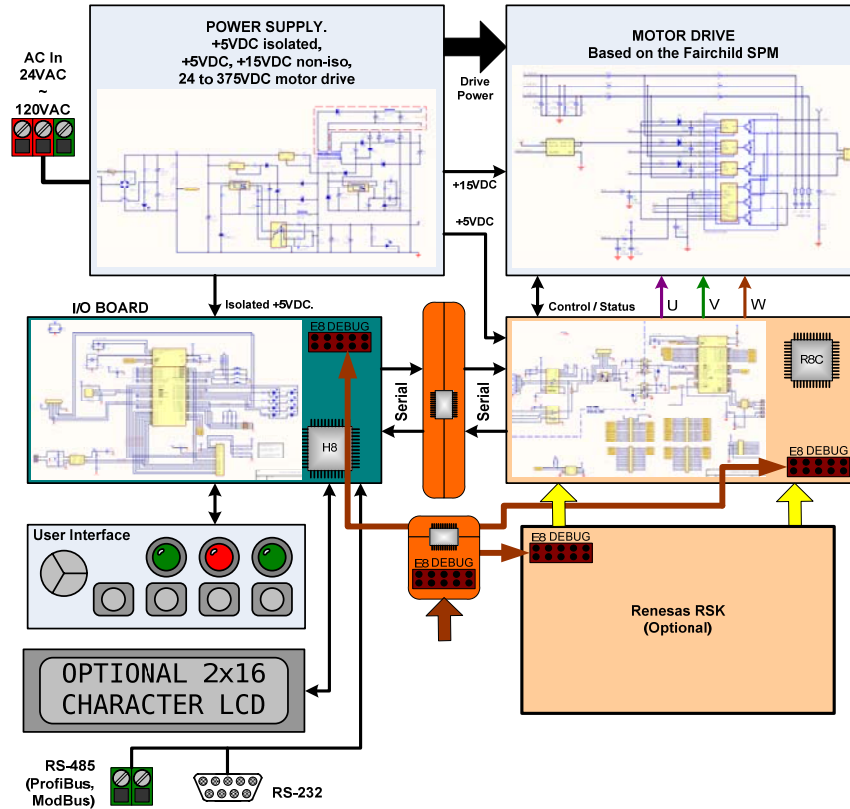
Thank you for your interest in All American's Quick-Spin Motor Controller kit and workshop. We've designed the kit and it's materials to aide today's busy engineer in "Quickly" "Spinning" a motor controller design based on Renesas MCUs and Fairchild SPMs of their own. We hope you enjoy the kit!

1.1 Features.

- Control 3 phase, AC Induction or Brushless DC (BLDC) motors from 24V to 220V, up to 3 Amps.
- Wide input power supply. 24 – 110 volts, AC or DC.
- EMI filtered, Hall Sensor Inputs.
- Built in isolated serial port with RS-232 and RS-422/RS-485 stuff option.
- Separate Motor Control and Communication MCUs allow for co-development of motor algorithms and communication protocols.
- Support option for Renesas RSKs.
- Built in isolated Renesas E8 debugger interface.
- In system programming for R8C and H8 MCUs.
- Isolated User interface board.
- Fully programmable user buttons, LEDs, Pot, and optional Encoder.
- LCD interface option.
- CAN interface option.
- Radiotronix wireless UART interface option.
- On board temperature sensors.

1.2 Block Diagram.

Block Diagram of the All American Quick-Spin system.



1.3 Technical Support.

For technical support, please contact your local All American FAE. You may also email MCUexperts@allamerican.com. When requesting technical support, please include the following information:

- Has the Main board or User I/O board software been modified.
- What method of control is being used for the board. I.e: Windows GUI, Text Menu, Hardware controls, Custom software, etc.
- Jumper settings.
- Input voltage.
- Type of motor connected.
- HEW version.
- A detailed description of the problem or issue.

2. Getting Started.

2.1 Kit Contents.

Your Quick-Spin kit contains the following items:

- Quick-Spin Main & User I/O board.
- Motor Control workshop manual.
- Motor Control workshop CD. (In rear cover of manual)

2.2 System Requirements.

The minimum hardware and software requirements are:

- PC running Windows® 2000/XP or higher.
- HyperTerminal or some other form of terminal program.
- 115200 baud RS-232 port. (COM port)
- 24V – 110V, AC or DC power supply, 1.25A minimum.
- 3 phase Brushless DC (BLDC) or AC Induction motor sized with power supply.
- Serial cable.
- Renesas E8 debugger. (Recommended.)
- Isolation transformer. (Recommended for high voltage development.)

2.3 Quick Start.

The Quick-Spin kit is shipped with default code to control a small, 24V, BLDC motor. However, this code may have been modified if the kit was used during the All American Motor Control workshop. In such case, the default code and hex images may be found on the CD that accompanies the kit.

For detailed information on how to connect and start using the Quick-Spin system, refer to Lab 1 in the Workshop Manual and CD that accompanies the kit.

3. Hardware.

3.1 Main Board.

This section describes various points of the Quick-Spin's Main board.

3.1.1 Power Supply.

To facilitate a wide range of motors as well as isolated and non-isolated circuits, the Quick-Spin board's power supply is highly versatile with the following features:

- Wide input voltage range from 24 to 110 volts, AC or DC.

- Built in voltage doubler for high voltage operation.
- Non Isolated Outputs:
 - +24VDC to +300VDC bus voltage. (depending on input voltage and jumper configuration) used to drive motors.
 - +15VDC used for IGBT Module gate drive.
 - +5VDC used for Motor control MCU and related electronics.
- Isolated Output:
 - +5VDC used for Communication MCU, User I/O board, and related electronics.

3.1.2 Motor Drive.

The Fairchild SPM (Smart Power Module) is the heart of the motor drive. The SPM contains 6 IGBTs to drive 3 phase motors up to 3 Amps. The SPM is supplied bus voltage from the board's power supply. It also receives a +15VDC gate voltage which is controlled via a FET switch external to the SPM. This switch allows the MCU to safely power down the SPM.

Bus and leg voltage is read via the MCU. Bus current is obtained via a current shunt, op amp, and read by the MCU.

3.1.3 MCU.

Performing the Motor Control functions is the Renesas R8C MCU. This MCU generates complementary, 3 phase drives signals for the Fairchild SPM. In addition, it handles all motor control calculations and operations which include the reading of BLDC hall sensors, bus current and voltage A/D conversion, drive and ambient temperature reading, control of the SPM's gate drive voltage, and communication to the system's isolated Communication MCU.

Note, unused pins on the R8C MCU are routed out to SMT pads on the top side of the Main board for ease in prototyping.

3.1.4 Isolated Serial Port.

The Main board contains an isolated serial port for communication with off board control software. RS-232 is the port's default standard with stuff options for RS-485 and RS-422. See the '*Jumpers and Stuff Options*' section for details on switching to RS-485/RS-422.

With the User I/O board removed from the Main board, the Motor Control MCU communicates across the isolation bridge to the serial transceiver. (RS-232 default.) However, with the User I/O board installed, the Motor Control MCU communicates across the isolation bridge with the Communication MCU on the User I/O board, which in turn, communicates with the serial transceiver.

This switching between which MCU communicates with the outside world is automatic when the User I/O board is installed or removed.

3.1.5 Temperature Sensors.

Drive temperature is obtained via a thermistor placed on the heatsink and read via the MCU's ADC.

Ambient temperature is obtained from U102 and fed back to the MCU via I²C bus.

3.1.6 MD100. (Radiotronix Module.)

The Main board has stuff options for a Radiotronix Wi.232DTS EVM module in the event the user wants to experiment with a Radiotronix wireless UART and the isolation it provides. See the '**Jumpers and Stuff Options**' section for details on populating the MD100 module.

3.1.7 LEDs.

There are 3 LEDs on the Main board. Their operation is as follows:

LED	Color	Function
D200	Red	Indicates +15VDC is being supplied to the SPM. (Gate drive is enabled.)
D300	Red	Indicates Bus Voltage is present.
D301	Green	Indicates +15VDC, non isolated power is present.

3.1.8 Connectors.

The following tables detail the connectors on the Quick-Spin Main board:

Connector: CN300		Name: Input Power
Pin	Signal	
1	±24 ~ 110 V AC/DC input.	
2	±24 ~ 110 V AC/DC input.	
3	Earth Ground.	

Connector: CN200		Name: Motor Drive
Pin	Signal	
1	W Phase Output.	
2	V Phase Output.	
3	U Phase Output.	
4	Earth Ground.	

Connector: CN109		Name: Hall Sensors
Pin	Signal	
1	Non-Isolated Ground.	
2	Non-Isolated +5VDC for Hall Sensors Power. (200mA max.)	
3	U Phase Hall Sensor Input.	
4	V Phase Hall Sensor Input.	
5	W Phase Hall Sensor Input.	

Connector: CN101		Name: RS-232 Port
Pin	Signal	
1	No Connection.	
2	Tx Data.	
3	Rx Data.	
4	Hardwire to Pin 6. (Handshake)	
5	Isolated Ground.	
6	Hardwire to Pin 4. (Handshake)	
7	Hardwire to Pin 8. (Handshake)	
8	Hardwire to Pin 7. (Handshake)	
9	No Connection.	

Connector: CN100		Name: RS-485/RS-422 Port
Pin	Signal	
1	+ Rx Data.	
2	- Rx Data.	
3	- Tx Data.	
4	+ Tx Data.	

Connector: CN102		Name: General Purpose Isolated I/O
Pin	Signal	
1	Isolated +5VDC.	
2	Isolated output from R8C.	
3	Isolated input to R8C.	
4	Isolated Ground.	

NOTE: CN102 is available for general use and has no current function.

Connector: CN103		Name: User I/O board connector
Pin	Signal	
1	Isolated +5VDC.	
2	Downstream Serial Rx.	
3	Downstream Serial Tx.	
4	User I/O Board Present.	

5	Upstream Serial Tx.
6	Upstream Serial Rx.
7	Isolated Ground.

Connector: CN108		Name: User I/O board connector	
Pin	Signal	Pin	Signal
1	E8 SCLK.	2	Non-Isolated Ground.
3	E8 CNVss.	4	E8 EPM.
5	E8 TxD.	6	Non-Isolated Ground.
7	E8 CE.	8	Non-Isolated +5VDC.
9	E8 Busy.	10	Non-Isolated Ground.
11	E8 RxD.	12	Non-Isolated Ground.
13	nReset.	14	Non-Isolated Ground.

Connector: CN104		Name: RSK Header	
Pin	Signal	Pin	Signal
1	Non-Isolated +5VDC.	2	Non-Isolated Ground.
3	Pad. (RSK 3V3.)	4	Non-Isolated Ground.
5	Pad. (RSK AVCC.)	6	Pad. (RSK AVSS.)
7	Pad. (RSK AVREF.)	8	ADTRG. (Hall V Input)
9	Bus Current. (AD0 & AD1)	10	Bus Current. (AD0 & AD1)
11	AD2.	12	Pad. (RSK DAC1.)
13	Pad. (RSK DAC0.)	14	IGBT Enable.
15	Thermistor.	16	Pad. (RSK IO_1.)
17	Pad. (RSK IO_2.)	18	Pad. (RSK IO_3.)
19	Pad. (RSK IO_4.)	20	Pad. (RSK IO_5.)
21	Pad. (RSK IO_6.)	22	Pad. (RSK IO_7.)
23	IRQ3. (Hall W Input)	24	Pad. (RSK I ² C EX.)
25	I ² C SDA.	26	I ² C SCL.

Connector: CN105		Name: RSK Header	
Pin	Signal	Pin	Signal
1	nReset	2	Pad. (RSK EXTAL.)
3	Pad. (RSK NMIn.)	4	Pad. (RSK Vss1.)
5	Pad. (WDT OVF.)	6	TxD0.
7	IRQ0. (IGBT Fault)	8	RxD0.
9	IRQ0. (Hall U Input)	10	Pad. (RSK SClACK.)
11	Pad. (RSK MO_UD.)	12	Pad. (RSK CTS/RTS.)
13	PWM Output, Up.	14	PWM Comp. Output, Un.
15	PWM Output, Vp.	16	PWM Comp. Output, Vn.
17	PWM Output, Wp.	18	PWM Comp. Output, Wn.
19	Pad. (RSK TMR0.)	20	Pad. (RSK TMR1.)
21	Pad. (RSK TRIGa.)	22	Pad. (RSK TRIGb.)
23	Pad. (RSK IRQ2.)	24	Pad. (RSK TRISTn.)
25	Pad. (RSK No Connect.)	26	Pad. (RSK No Connect.)


Connector: CN106		Name: RSK Header	
Pin	Signal	Pin	Signal
1	Bus Volts. (AD4.)	2	U Phase Volts. (AD5.)
3	V Phase Volts. (AD6.)	4	W Phase Volts. (AD7.)

5	Pad. (RSK CAN1TX.)	6	Pad. (RSK CAN1RX.)
7	Pad. (RSK CAN2TX.)	8	Pad. (RSK CAN2RX.)
9	Pad. (RSK No Connect.)	10	Pad. (RSK No Connect.)
11	Pad. (RSK No Connect.)	12	Pad. (RSK No Connect.)
13	Pad. (RSK No Connect.)	14	Pad. (RSK No Connect.)
15	Pad. (RSK No Connect.)	16	Pad. (RSK No Connect.)
17	Pad. (RSK No Connect.)	18	Pad. (RSK No Connect.)
19	Pad. (RSK No Connect.)	20	Pad. (RSK No Connect.)
21	Pad. (RSK No Connect.)	22	Pad. (RSK No Connect.)
23	Pad. (RSK No Connect.)	24	Pad. (RSK No Connect.)

Connector: CN107		Name: RSK Header	
Pin	Signal	Pin	Signal
1	Pad. (RSK DREQ.)	2	Pad. (RSK DACK.)
3	Pad. (RSK TEND.)	4	Pad. (RSK STBYn.)
5	Pad. (RSK RS232TX.)	6	Pad. (RSK RS232RX.)
7	Pad. (RSK SC1bRX.)	8	Pad. (RSK SC1bTX.)
9	Pad. (RSK SC1cTX.)	10	Pad. (RSK SC1bCK.)
11	Pad. (RSK SC1cCK.)	12	Pad. (RSK SC1cRX.)
13	Pad. (RSK No Connect.)	14	Pad. (RSK No Connect.)
15	Pad. (RSK No Connect.)	16	Pad. (RSK No Connect.)
17	Pad. (RSK No Connect.)	18	Pad. (RSK No Connect.)
19	Pad. (RSK No Connect.)	20	Pad. (RSK No Connect.)
21	Pad. (RSK No Connect.)	22	Pad. (RSK No Connect.)
23	Pad. (RSK No Connect.)	24	Pad. (RSK No Connect.)

3.1.9 Jumpers and Options.

The following tables list the various options and jumper settings for the Quick-Spin Main board.

Jumper: J300	Name: High Voltage Jumper
Open:	AC/DC operation under 50V. Voltage doubler disabled.
Closed:	AC/DC operation above 50V. Voltage doubler enabled.
 <p>WARNING!!! In this mode, INPUT VOLTAGE IS DOUBLED. ie: At 120V in, Bus Voltage is in excess of 240V!! <u>FAILURE TO OBSERVE PROPER SAFETY MEASURES COULD RESULT IN SEVERE ELECTRICAL SHOCK, OR EVEN DEATH.</u></p>	

NOTE: To close J300, solder in 18 gauge wire or better.

Jumper: J100	Name: R8C Reset / Tristate
Open:	Normal R8C (U106) operation.
Closed:	R8C (U106) is in reset and all pins are tri-stated. When using an RSK with the Quick-Spin board, this jumper MUST BE CLOSED to keep the onboard R8C in a tristate condition.

Serial Port Stuff Options. (NOTE: May require additional user programming.)		
Option	Stuff	No Stuff
RS-232	R119, R121	R118, R120
RS-485 / RS-422	R120, U101, C103, CN100	R118, R119
Wireless UART	MD100, C105, R118	R121, User I/O board

NOTE: See BOM and schematics for component values and part numbers.

RSK Stuff Option. (NOTE: May require additional user programming.)		
Option	Stuff	No Stuff
To use an RSK instead of the onboard R8C	CN104, CN105, CN106, CN107 Close J100.	Remove RSK and J100 for normal operation.

NOTE: Any RSK that supports motor control may be used. RSK must face up with pin headers on bottom. See BOM for details on CN104-CN107. If correct parts are not used for CN104-CN107, boards will not stack correctly.

Headless Operation.		
Option	Headless	With User I/O board
To use the Quick-Spin board without a User I/O board.	Remove the User I/O board from the Main board.	Plug the User I/O board back onto the Main board.

NOTE: In this mode, the Main board cannot be debugged using the E8 Debugger.

3.2 User I/O Board.

This section describes various points of the Quick-Spin's User I/O board.

3.2.1 E8 Debug Interface.

To facilitate debugging, the User I/O board contains an E8 Debug port. This port can be used for the H8 Communication MCU on the User I/O board, the R8C Motor Control MCU on the Main board, or an optional RSK fitted to the Main board. A simple jumper change is all that is needed to switch between MCUs. See the '**Jumpers and Stuff Options**' section for the **User I/O board** for details on jumper settings.

3.2.2 Debug Isolation Bridge.

A bi directional, optical isolation bridge for the E8 debug interface is built into the User I/O board. This isolation bridge is used to communicate with either the Main board's R8C MCU (U106) or an optional RSK fitted to the Main board. Since either of these devices operate using non-isolated voltages, the built in isolation bridge provides protection to external E8 hardware, Host PCs, etc.

Noted that the User I/O board is INTENTIONALLY DESIGNED to interfere with a User plugging into the various ports on RSK fitted to the Main board, including the RSK's E8 port. This again, is because the RSK operates referenced to the input voltage and is considered 'HOT'. **Bypassing mechanical or electrical isolation features may result in equipment or component damage to both the Quick-Spin system and connected external devices.**

3.2.3 MCU.

Performing communications functions is the Renesas H8 MCU. This MCU is provided to allow separate development of high level control protocols such as Modbus, Profibus, and CAN without interfering with development of motor control algorithms. This power MCU features a CAN interface, timers, UARTs, etc. It also bolts up to several user interface components such as LEDs, buttons, a pot, optional encoder and LCD.

This Communications MCU is on the isolated side of the Quick-Spin system and communicates with the Main board's MCU or fitted RSK via an isolated serial port. See the '***Isolated Serial Port***' section for the ***Main board*** for more information.

Note, unused pins on the H8 MCU are routed out to SMT pads on both sides of the User I/O board for ease in prototyping.

3.2.4 LEDs.

There are 4, fully programmable, dual color (red and green) LEDs on the User I/O board. Their default operation is as follows:

1. On startup, the User I/O board's MCU establishes communication with the Main board's MCU. During this time, each LED is turned on for .5 sec according to the following sequence:
 - a. Green D5 -> D6 -> D7 -> D8 ->
 - b. Red D8 -> D7 -> D6 -> D5
 - c. Repeat full sequence until connected.
2. Once the boards have connected, D7 Red turns on to indicate that the motor is stopped. All other LEDs are off.
3. D5 Red turns on in case of a motor fault. D5 Red and Green (Yellow) are on if the fault is old. The direction LEDs will remain on in the state of the current direction.
4. D6 Green turns on when the motor is moving in the reverse direction. D5 remains in the same state and all other LEDs are off.

5. D7 Red turns on when the motor is stopped. D5 remains in the same state and all other LEDs are off.
6. D8 Green turns on when the motor is running forward. D5 remains in the same state and all other LEDs are off.

3.2.5 Buttons.

There are 4 fully programmable buttons on the User I/O board. Their default operation is as follows:

Button	Default Operation
SW3	Clear Motor Fault(s).
SW4	Motor Reverse.
SW5	Stop.
SW6	Motor Forward.

3.2.6 Pot.

An analog pot is connected to an input of the H8 Communication MCU's ADC. The pot can be programmed for any use with it's default being that of a speed control.

3.2.7 Connectors.

The following tables detail the connectors on the Quick-Spin User I/O board:

Connector: CN1		Name: E8 Debug connector	
Pin	Signal	Pin	Signal
1	E8 SCLK.	2	Isolated Ground.
3	E8 CNVss.	4	E8 EPM.
5	E8 TxD.	6	Isolated Ground.
7	E8 CE.	8	Isolated +5VDC.
9	E8 Busy.	10	Isolated Ground.
11	E8 RxD.	12	Isolated Ground.
13	E8 RESET.	14	Isolated Ground.

Connector: CN2		Name: Main board Debug	
Pin	Signal	Pin	Signal
1	Non-Isolated E8 SCLK.	2	Non-Isolated Ground.
3	Non-Isolated E8 CNVss.	4	Non-Isolated E8 EPM.
5	Non-Isolated E8 TxD.	6	Non-Isolated Ground.
7	Non-Isolated E8 CE.	8	Non-Isolated +5VDC.
9	Non-Isolated E8 Busy.	10	Non-Isolated Ground.
11	Non-Isolated E8 RxD.	12	Non-Isolated Ground.
13	Non-Isolated E8 RESn.	14	Non-Isolated Ground.

Connector: CN3		Name: RSK E8 Debug	
Pin	Signal	Pin	Signal
1	Non-Isolated E8 SCLK.	2	Non-Isolated Ground.
3	Non-Isolated E8 CNVss.	4	Non-Isolated E8 EPM.
5	Non-Isolated E8 TxD.	6	Non-Isolated Ground.
7	Non-Isolated E8 CE.	8	Non-Isolated +5VDC.
9	Non-Isolated E8 Busy.	10	Non-Isolated Ground.
11	Non-Isolated E8 RxD.	12	Non-Isolated Ground.
13	Non-Isolated E8 RESn.	14	Non-Isolated Ground.

Connector: CN4		Name: CAN Port	
Pin	Signal		
1	+ CAN Data.		
2	- CAN Data.		

Connector: CN5		Name: User I/O board connector	
Pin	Signal		
1	Isolated +5VDC.		
2	Downstream Serial Rx.		
3	Downstream Serial Tx.		
4	User I/O Board Present. (Isolated Ground pulldown)		
5	Upstream Serial Tx.		
6	Upstream Serial Rx.		
7	Isolated Ground.		

Connector: LCD1		Name: LCD Option	
Pin	Signal		
1	± LED backlight. (See 'User I/O-Jumpers and Options' section for more info.)		
2	± LED backlight. (See 'User I/O-Jumpers and Options' section for more info.)		
3	Isolated Ground.		
4	Isolated +5VDC.		
5	LCD contrast.		
6	RS.		
7	R/W.		
8	E.		
9	Data 0.		
10	Data 1.		
11	Data 2.		
12	Data 3.		
13	Data 4.		
14	Data 5.		
15	Data 6.		
16	Data 7.		

NOTE: The LCD is a stuff option and requires additional components to be stuffed. See the 'User I/O – Jumpers and Options' section for more info.

Connector: J2		Name: Set Point Jumpers	
Pin	Signal	Pin	Signal

1	Set Point Input 0.	2	Isolated Ground.
3	Set Point Input 1.	4	Isolated Ground.
5	Set Point Input 2.	6	Isolated Ground.
7	Set Point Input 4.	8	Isolated Ground.

NOTE: Set Point Jumpers J2 is currently unused and may be programmed as desired. See the 'User I/O – Jumpers and Options' section for more info.

3.2.8 Jumpers and Options.

The following tables list the various options and jumper settings for the Quick-Spin User I/O board.

Jumper: J1	Name: E8 Debug Selector
Open:	Normal operation of all MCUs.
1 – 2 Closed:	Isolated debug of the Main board's R8C (U106) MCU or an optional RSK's MCU fitted to the Main board.
2 – 3 Closed:	Non-Isolated debug of the User I/O board's H8 MCU.

Encoder Stuff Option. (NOTE: Requires additional user programming.)	
Option	Stuff
Addition of a 2 bit Encoder control with push button.	SW1, R28, R29, R30.

NOTE: See BOM and schematics for component values and part numbers.

CAN Port Stuff Option. (NOTE: Requires additional user programming.)	
Option	Stuff
Addition of optional CAN port.	CN4, U7, R31, R32, R33, R34, C8, C12.

NOTE: See BOM and schematics for component values and part numbers.

LCD Stuff Option. (NOTE: Requires additional user programming.)		
Option	Stuff	
Addition of optional LCD.	LCD1, R36, backlight option...	
	LCD backlight, Pin1 positive / Pin2 negative:	R40 and R43.
	LCD backlight, Pin1 negative / Pin2 positive:	R42 and R41.

NOTE: See BOM and schematics for component values and part numbers. See 'User I/O – Connectors' for LCD pinout.

Set Points Stuff Option. (NOTE: Requires additional user programming.)	
Option	Stuff
Addition of Set Point Jumpers. Allows hardware control of up to 16 Set Points.	J2.

NOTE: J2 consists of General Purpose I/O connected to a header with opposing ground connections. This setup could be used as a 1 of 16 input to the Communications MCU to provide

a hardware control of 16 different motor speed Set Points. This feature is currently not implemented. See BOM and schematics for details and component part number.

4. Embedded Software.

4.1 Main Board R8C Files.

The following list briefly describes the C files for the Main board's MCU. Note, this list does not includes various files which are automatically generated by HEW.

File Name	Purpose
Defines.h	Contains system constants and #defines.
Adc.h / .c	Contains ADC initialization code.
Cmd_Proc.h / .c	Contains the command processor which handles all serial commands. (GUI and Text Menu.)
Motor_Variable_definition.h	Contains variables used by, and to, control the motor's operation.
ProtocolB_menus.h / .c	Contains text menu functions and related code.
Uart.h / .c	Contains UART driver code.
Main.c	Main startup code for the Motor Control MCU.
I2c.h / .c	I2C driver code.
Motor.h / .c	Contains the core motor control algorithm code.

4.2 I/O Board H8 Files.

The following list briefly describes the C files for the User I/O board's MCU. Note, this list does not includes various files which are automatically generated by HEW.

File Name	Purpose
Defines.h	Contains system constants and #defines.
Leds.h / .c	Contains code to control and scan the board's LEDs.
Cmd_Proc.h / .c	Contains the command processor which handles all serial commands. (GUI and Text Menu.)
Motor_Variable_definition.h	Contains variables used by, and to, control the motor's operation.
ProtocolB_menus.h / .c	Contains text menu functions and related code.
Uart.h / .c	Contains UART driver code.
Main.c	Main startup code for the Communications MCU.

5. Communications Software Specification.

5.1 Overview.

The Quick-Spin GUI and Communication specification, hereafter referred to as the 'spec' provides information on the following:

- 1- The communication protocols used by the Quick-Spin motor control board's Renesas microcontroller. There are two separate protocols:
 - a. A terse (menu-less) protocol designed for use with other processing systems such as a mater MCU, controller, or GUI. This protocol operates through commands that are not formatted for human readability.
 - b. A verbose, ASCII text based protocol designed for use with terminal communications or systems. This protocol is preformatted with simple menu and command prompts making it easily human readable.
- 2- A PC based GUI interface using *Protocol A* as it's communication method. Designed for use with Windows 2000 and Windows XP using Visual Studio 6.0, the GUI provides an open source framework that may be incorporated into user applications as well as an example control surface for the Quick-Spin system.

5.2 Communication Protocols.

5.2.1 Communication Medium.

The Quick-Spin board communicates primarily through two different methods. In both cases, the physical portion of the protocol shall comply with common standards allowing the use of serial to USB converters if necessary.

- 1- A 2 wire, RS-232 serial interface consisting of Tx, Rx, and Gnd.
- 2- A 4 wire, RS-422/485 serial interface consisting of +Tx/-Tx, +Rx/-Rx, and Gnd.

A CAN interface is implemented as an option for future development.

5.2.2 Communication Protocols Overview.

Both protocols are based solely on ASCII text. All commands and variables, including those in hex, are passed as ASCII text character thus simplifying the implementation of either protocol.

Both protocols use the same command characters, however, Protocol A is designed to be used as a program controlled interface in conjunction with a microcontroller, microprocessor, or other intelligent managing system. Each command with Protocol A must be preceded with a dollar sign '\$'. Command variables with Protocol A are sent directly following the command and upon reception and successful processing of a command, the Quick-Spin board return an exclamation mark '!' as an ACK character. The GUI that comes with the Quick-Spin system uses this protocol.

Protocol B is used in conjunction with terminal interfaces and programs. Protocol B is a complete text based menu system that will provide system variables and status and prompt the user for input based on command selections.

Upon power up, the Quick-Spin board initializes it's communication ports and waits for a character. If the first character received is a dollar sign, '\$', the Quick-Spin board automatically selects Protocol A as it's communication protocol. If the first character is not a dollar sign, Protocol B is used and the Quick-Spin board outputs it's Main Menu to the communication port.

At any time during Protocol B, the '@ @' command will force a switch to Protocol A.

5.2.3 Unused ASCII Characters.

The following characters are reserved and ignored in both protocols. Whitespace is also always ignored.

" (double quote)	3	o	o
#	4	Q	q
%	5	[r
&	6	\	t
' (grave)	7]	u
(8	^	v
)	9	_ (underscore)	w
*	: (colon)	` (apostrophe)	x
, (comma)	; (semicolon)	f	z
. (period)	<	g	{
/	=	h	(pipe)
0	>	l	}
1	H	m	~ (tilde)
2	J	n	

5.2.4 Protocol Commands at a Glance.

Motor Use	Command		Definition
	Protocol A	Protocol B	
Both	E	E	Emergency Stop. (Minimum Decel time.)
Both	K	K	Coast to Stop. (Kill PWM.)
Both	S	S	Controlled Stop with Decel.
AC only	X	X	Stop with DC Injection Brake.
Both	F	F	Forward at set Speed.
Both	R	R	Reverse at set Speed.
Both	N	N	Set New Speed. (In Hz or RPM.)
Both	+	+	Increase Speed.
Both	-	-	Decrease Speed.
Both	Y or y	Y or y	Status Info.
Both	n/a	?	Show this Menu.
Both	M	M	Set number of Motor Poles.
Both	U	U	Set Upper Speed Limit. (Maximum speed in RPM.)
Both	L	L	Set Lower Speed Limit. (Minimum speed in RPM.)
Both	T	T	Set Accel Time in Seconds. (0mS to 25.5 sec in 100mS increments.)
Both	V	V	Set Decel Time in Seconds. (0mS to 25.5 sec in 100mS increments.)
AC only	B	B	Set Boost Voltage.
AC only	W	W	Set DC Injection Brake Voltage.
DC only	n/a	G	Show Speed Gain PID Menu.
DC only	P	P	Proportional Gain. (Speed gain.)
DC only	I	I	Integral Gain. (Speed gain.)
DC only	D	D	Differential Gain. (Speed gain.)
DC only	p	p	Proportional Gain. (Current gain.)
DC only	i	i	Integral Gain. (Current gain.)
DC only	d	d	Differential Gain. (Current gain.)
Both	j	j	Mode of Operation. (Open Loop, Speed PID w/ 120° hall, Speed PID w/ 60° hall, One Shunt., etc.)
Both	k	k	Motor Select. (AC Induction or BLDC Motor.)
Both	a	a	Rated Motor Frequency in hertz.
Both	b	b	Rated Motor Voltage in volts.
Both	A	n/a	Get Set Points. (Returns all Set Points to the GUI.)
Both	C	n/a	Save Set Point. (Save the current info as set point 'n'.)
Both	e	n/a	Load Set Point.
Both	Z	Z	Reset the board.
Both	c	c	Clear current Fault Flags.
Both	n/a	s	Show Motor Parameters Menu.

5.3 Protocol A Command Set.

This section details the various commands and data strings for Protocol A which are sent to, and received from, the Quick-Spin board.

Protocol notes:

- Commands are shown enclosed with single quotes ‘ ’ for clarity only. Single quotes are not actually used.
- Whitespace in between characters is added for clarity only and is not required. Any whitespace in commands is ignored.
- All values are sent in hex as ASCII characters. For example:
 - The number 23 is hex 0x17 and would be sent as ‘1’ (0x31), ‘7’ (0x37).
 - Hex value 0xA5 would be sent as ‘A’ (0x41) and ‘5’ (0x35)
 - The number 175 is hex 0xAF and would be sent as ‘A’ (0x41), ‘F’ (0x45).
- The Quick-Spin board returns an exclamation point ‘!’ as an ACK after each command provided that command was successful. Noting is returned if the command fails.

5.3.1 Protocol A Command Details.

Command:	Reset	
ASCII String:	‘\$ Z’	
Parameters Required:	2 characters	
Total Tx Characters:	4	
Description:	Resets the Quick-Spin board to a known and ready state. <ul style="list-style-type: none"> • Motor stopped. • Drive system disabled. (PWM offline) • Dynamic motor variables reset to default. 	
Parameters:	Value	Parameter Description:
1 & 2	‘O K’	These two safety characters tell the Quick-Spin it’s really ok to reset. Without these, the reset command is ignored.
Returns:	#	<ul style="list-style-type: none"> • ‘#’ Hash (Pound) indicates the start of a string sent from the Quick-Spin board that is something other than ‘!’ (ACK).
	R	<ul style="list-style-type: none"> • ‘R’ – Indicates a reset command has been received and is being processed.
	!	<ul style="list-style-type: none"> • ‘!’ – ACK.

Command:	Emergency Stop	
ASCII String:	‘\$ E’	
Parameters Required:	None	
Total Tx Characters:	2	
Description:	Stops the motor using the minimum deceleration time.	
Parameters:	Value	Parameter Description:

None		
Returns:	' ! '	ACK if successful.

Command:	Coast to Stop	
ASCII String:	' \$ K '	
Parameters Required:	None	
Total Tx Characters:	2	
Description:	Stops the motor by providing zero drive and allowing the motor to mechanically coast until motionless.	
Parameters:	Value	Parameter Description:
	None	
Returns:	' ! '	ACK if successful.

Command:	Controlled Stop using Deceleration.	
ASCII String:	' \$ S '	
Parameters Required:	None	
Total Tx Characters:	2	
Description:	Stops the motor using the system's programmed deceleration time.	
Parameters:	Value	Parameter Description:
	None	
Returns:	' ! '	ACK if successful.

Command:	Stop using DC Injection Braking. (AC motors only)	
ASCII String:	' \$ X '	
Parameters Required:	None	
Total Tx Characters:	2	
Description:	Stops an AC motor using a DC injection braking method. NOTE: For this command to operate successfully, the DC Injection Brake Voltage must be set.	
Parameters:	Value	Parameter Description:
	None	
Returns:	' ! '	ACK if successful.

Command:	Forward Start	
ASCII String:	' \$ F '	
Parameters Required:	None	
Total Tx Characters:	2	
Description:	<ul style="list-style-type: none"> Starts the motor in a forward direction. NOTE: For this command to operate correctly, the system must be clear of any fault conditions. 	
Parameters:	Value	Parameter Description:
	None	
Returns:	' ! '	ACK if successful.

Command:	Reverse Start	
ASCII String:	' \$ R '	
Parameters Required:	None	
Total Tx Characters:	2	

Description:	Starts the motor in a reverse direction. NOTE: For this command to operate correctly, the system must be clear of any fault conditions.	
Parameters:	Value	Parameter Description:
	None	
Returns:	' ! '	ACK if successful.

Command:	Set Speed	
ASCII String:	'\$ N'	
Parameters Required:	4 characters sent as a single 16 bit hex value	
Total Tx Characters:	6	
Description:	Sets a new motor speed in RPM. If the drive is in operation, the motor shall be accelerated or decelerated to this new speed using the programmable acceleration or deceleration times. If the drive is not in operation, the speed value is changed but no motion results. The drive does not become active or inactive through the use of this command. If the value sent with this command is beyond (above or below) the Minimum or Maximum speed limits, the value is replaced with that limit.	
Parameters:	Value	Parameter Description:
	1	0 – FFFF
		A hex value from 0 to FFFF indicating motor speed in RPM. NOTE: The speed value is subject to the minimum and maximum speed settings. Speed settings beyond the minimum and maximum limits cause the speed setting to become the minimum or maximum limit
Returns:	' ! '	ACK if successful.

Command:	Increase Speed	
ASCII String:	'\$ +'	
Parameters Required:	1 character sent as a hex nibble.	
Total Tx Characters:	3	
Description:	Increases the current speed by 1 of 8 values as defined in the following parameter. The command ceases to increase or decrease the speed value once the maximum or minimum programmed speed is reached. If the drive is faulted, or in stop mode, this command is ignored.	
Parameters:	Value	Parameter Description:
	1	0 – 7
		A number from 0 to 7 indicating 1 of the 8 following increment values. NOTE: Values 8 to 0xF are ignored. 0- Default. Increment by 1. 1- Increment by 2. 2- Increment by 5. 3- Increment by 10. 4- Increment by 25. 5- Increment by 50. 6- Increment by 100. 7- Increment by 200.
Returns:	' ! '	ACK if successful.

Command:	Decrease Speed	
ASCII String:	'\$ -'	

Parameters Required:	1 character sent as a next nibble.	
Total Tx Characters:	3	
Description:	Decreases the current speed by 1 of 8 values as defined in the following parameter. The command ceases to decrease the speed value once the minimum programmed speed is reached. If the drive is faulted, or in stop mode, this command is ignored.	
Parameters:	Value	Parameter Description:
1	0 – 7	A number from 0 to 7 indicating 1 of the 8 following decrement values. NOTE: Values 8 to 0xF are ignored. 0- Default. Decrement by 1. 1- Decrement by 2. 2- Decrement by 5. 3- Decrement by 10. 4- Increment by 25. 5- Increment by 50. 6- Increment by 100. 7- Increment by 200.
Returns:	' ! '	ACK if successful.

Command:	Get Status	
ASCII String:	' \$ Y ' –OR- ' \$ y '	
Parameters Required:	None	
Total Tx Characters:	2	
Description:	Requests the STATUS string from the Quick-Spin board.	
Parameters:	Value	Parameter Description:
None		
Returns:		This command returns the following STATUS string.
81 characters.	#	<ul style="list-style-type: none"> ' # ' Hash (Pound) indicates the start of a string sent from the Quick-Spin board that is something other than ' ! ' (ACK).
	'S'	<ul style="list-style-type: none"> The 'S' indicates the following is a Status string.
	00 – FF	<ul style="list-style-type: none"> 2 characters as a hex byte of Fault indicators. 1- Fault, 0- No fault. <ul style="list-style-type: none"> 0- Over Voltage. 1- Under Voltage. 2- Over Current. 3- Short Circuit. 4- Over Temperature. 5- Hall Sensor Error. 6- Motor Stall. 7- Emergency Stop Loop. (1- Loop Open, 0- Loop Closed).
	00 – FF	<ul style="list-style-type: none"> 2 characters as a hex byte of Previous Fault indicators. The format is the same the Fault indicator byte.
	00 – FF	<ul style="list-style-type: none"> 2 characters as a hex byte of System Flags. <ul style="list-style-type: none"> 0- Motor Status, bit 0. 1- Motor Status, bit 1.

		<p>00- Motor off. 01- Motor spinning forward. 10- Motor spinning in reverse. 11- Reserved.</p> <p>2- Drive. 1- Drive enabled, 0- Drive disabled. 3- Hall Sensor 1. 1- Hall output, 0- No hall output. 4- Hall Sensor 2. 1- Hall output, 0- No hall output. 5- Hall Sensor 3. 1- Hall output, 0- No hall output. 6- Pending Motor Status, bit 0. 7- Pending Motor Status, bit 1.</p> <p>00- Motor off. 01- Motor spinning forward. 10- Motor spinning in reverse. 11- Reserved.</p>
	0000 – FFFF	<ul style="list-style-type: none"> • 4 characters as a 16 bit hex number returning programmed motor speed in RPM.
	0000 – FFFF	<ul style="list-style-type: none"> • 4 characters as a 16 bit hex number returning actual motor speed in RPM.
	000 – 3E7	<ul style="list-style-type: none"> • 3 characters as a 12 bit hex number indicating motor current in 100's of milliamps with the maximum valid motor current being 99.9 Amps or 0x3E7.
	000 – FFF	<ul style="list-style-type: none"> • 3 characters as a 12 bit hex number indicating DC Bus Voltage.
	00 – FFF	<ul style="list-style-type: none"> • 3 characters as a hex byte indicating drive temperature in degrees C from -2047 to +2047.
	00 – FFF	<ul style="list-style-type: none"> • 3 characters as a hex byte indicating ambient temperate in degrees C from -2047 to +2047.
	00 – FF	<ul style="list-style-type: none"> • 2 characters as a hex byte indicating the raw value of the Quick-Spin board's speed control pot.
	00 – 3F	<ul style="list-style-type: none"> • 2 characters as a hex byte indicating PWM phase status as below. NOTE: This rough indication of PWM phase is used only to create a graphical representation of the PWM waveforms. This data is not real time or accurate for any other purpose. <ul style="list-style-type: none"> ○ Bits 0 and 1 as drive leg U. ○ Bits 2 and 3 as drive leg V. ○ Bits 4 and 5 as drive leg W. ○ Bits 6 and 7 are reserved. ○ 00 – Leg off. (Not Driving.) ○ 01 – Leg is driving and is the 0° leg. ○ 10 – Leg is driving and is the 60° leg. ○ 11 – Leg is driving and is the 120° leg.
	0000 – FFFF	<ul style="list-style-type: none"> • 4 characters as a 16 bit hex value containing the Proportional gain setting of the Speed Gain PID.
	0000 – FFFF	<ul style="list-style-type: none"> • 4 characters as a 16 bit hex value containing the Integral gain setting of the Speed Gain PID.

0000 – FFFF	<ul style="list-style-type: none"> 4 characters as a 16 bit hex value containing the Differential gain setting of the Speed Gain PID.
0000 – FFFF	<ul style="list-style-type: none"> 4 characters as a 16 bit hex value containing the Proportional gain setting of the Current Gain PID.
0000 – FFFF	<ul style="list-style-type: none"> 4 characters as a 16 bit hex value containing the Integral gain setting of the Current Gain PID.
0000 – FFFF	<ul style="list-style-type: none"> 4 characters as a 16 bit hex value containing the Differential gain setting of the Current Gain PID.
0 – F	<ul style="list-style-type: none"> 1 character as a hex nibble containing the number of poles for the motor.
0000 – FFFF	<ul style="list-style-type: none"> 4 characters as a 16 bit hex value containing the Upper Speed Limit in RPM.
0000 – FFFF	<ul style="list-style-type: none"> 4 characters as a 16 bit hex value containing the Lower Speed Limit in RPM.
00 – FF	<ul style="list-style-type: none"> 2 characters as a hex byte indicating the Acceleration Time in 100's of milliseconds with a maximum value of 25.5 seconds.
00 – FF	<ul style="list-style-type: none"> 2 characters as a hex byte indicating the Deceleration Time in 100's of milliseconds with a maximum value of 25.5 seconds.
00 – FF	<ul style="list-style-type: none"> 2 characters as a hex byte representing the motor boost voltage in volts.
00 – FF	<ul style="list-style-type: none"> 2 characters as a hex byte representing the DC injection brake voltage in volts.
0 – F	<ul style="list-style-type: none"> 1 character as a hex nibble indicating the mode of operation. See the <i>Mode of Operation 'j'</i> command for more info.
0 – F	<ul style="list-style-type: none"> 1 character as a hex nibble indicating the motor selection. See the <i>Motor Type 'k'</i> command for more info.
00 – FF	<ul style="list-style-type: none"> 2 characters as a hex byte indicating the motor's Rated Frequency in hertz.
00 – FF	<ul style="list-style-type: none"> 2 characters as a hex byte indicating the motor's Rated Voltage.
!	<ul style="list-style-type: none"> '!' ACK character.

Command:	Set Number of Motor Poles				
ASCII String:	'\$ M'				
Parameters Required:	1 character sent as a hex nibble.				
Total Tx Characters:	3				
Description:	Sets the number of motor poles for the motor connected to the Quick-Spin board.				
Parameters:	<table border="1"> <thead> <tr> <th>Value</th> <th>Parameter Description:</th> </tr> </thead> <tbody> <tr> <td> </td> <td> </td> </tr> </tbody> </table>	Value	Parameter Description:		
Value	Parameter Description:				

1	0 – F	A hex value from 0 to F indicating the number of motor poles contained within the motor currently connected to the Quick-Spin board.
Returns:	' ! '	ACK if successful.

Command:	Set Upper Speed Limit. (in RPMs.)	
ASCII String:	' \$ U '	
Parameters Required:	4 characters sent as a single 16 bit hex value	
Total Tx Characters:	6	
Description:	Sets the upper speed limit value for the Quick-Spin board.	
Parameters:	Value	Parameter Description:
1	0 – FFFF	A hex value from 0 to FFFF indicating motor speed in RPM.
Returns:	' ! '	ACK if successful.

Command:	Set Lower Speed Limit. (in RPMs.)	
ASCII String:	' \$ L '	
Parameters Required:	4 characters sent as a single 16 bit hex value	
Total Tx Characters:	6	
Description:	Sets the lower speed limit value for the Quick-Spin board.	
Parameters:	Value	Parameter Description:
1	0 – FFFF	A hex value from 0 to FFFF indicating motor speed in RPM.
Returns:	' ! '	ACK if successful.

Command:	Set Acceleration Time in 100 Millisecond Increments.	
ASCII String:	' \$ T '	
Parameters Required:	2 characters sent as a hex byte.	
Total Tx Characters:	4	
Description:	Sets the acceleration time in 100 millisecond increments used for motor spin up and positive speed changes. The maximum acceleration time is 25.5 seconds. NOTE: An exception is the <i>Increase Speed</i> ' + ' command which uses its own set of values.	
Parameters:	Value	Parameter Description:
1	0 – FF	A hex value from 0 to FF indicating motor acceleration time in 100 millisecond increments.
Returns:	' ! '	ACK if successful.

Command:	Set Deceleration Time in 100 Millisecond Increments.	
ASCII String:	' \$ V '	
Parameters Required:	2 characters sent as a hex byte.	
Total Tx Characters:	4	
Description:	Sets the deceleration time in 100 millisecond increments used for motor wind down and negative speed changes. The maximum deceleration time is 25.5 seconds. NOTE: An exception is the <i>Decrease Speed</i> ' - ' command which uses its own set of values.	

Parameters:	Value	Parameter Description:
1	0 – FF	A hex value from 0 to FF indicating motor deceleration time in 100 millisecond increments.
Returns:	'!'	ACK if successful.

Command:		
Set Boost Voltage. (AC motors only)		
ASCII String:		
'\$ B'		
Parameters Required:		
2 characters sent as a hex byte.		
Total Tx Characters:		
4		
Description:	Sets the boost voltage variable in the V/F algorithm used in driving motors.	
Parameters:	Value	Parameter Description:
1	0 – FF	A hex value from 0 to FF indicating motor boost voltage in volts.
Returns:	'!'	ACK if successful.

Command:		
Set DC Injection Brake Voltage. (AC motors only)		
ASCII String:		
'\$ W'		
Parameters Required:		
2 characters sent as a hex byte.		
Total Tx Characters:		
4		
Description:	Sets the DC injection brake voltage for use in DC braking of AC motors.	
Parameters:	Value	Parameter Description:
1	0 – FF	A hex value from 0 to FF indicating the DC injection brake voltage in volts.
Returns:	'!'	ACK if successful.

Command:		
Mode of Operation		
ASCII String:		
'\$ j'		
Parameters Required:		
1 character sent as a hex nibble.		
Total Tx Characters:		
3		
Description:	Sets the Quick-Spin board's mode of operation using one of the values in the following parameter. NOTE: This command works in conjunction with the <i>Motor Type</i> 'k' command.	
Parameters:	Value	Parameter Description:
1	0 – F	A hex value from 0 to F indicating 1 of 16 possible operating mode for the Quick-Spin board. 0- Open loop. (Usually AC motors only.) 1- Speed PID using 120° hall sensors. (Quick-Spin Default.) 2- Speed PID using 60° hall sensors. 3- Speed & Current PID using 120° hall sensors. (Not currently implemented.) 4- Speed & Current PID using 60° hall sensors. (Not currently implemented.) 5- All American One Shunt. (Not currently implemented.) 6- Renesas One Shunt. (Not currently implemented.)

		Values 7 to F are reserved.
Returns:	'!'	ACK if successful.

Command:		Motor Type
ASCII String:		'\$ k'
Parameters Required:		1 character sent as a hex nibble.
Total Tx Characters:		3
Description:	Sets the Quick-Spin board for operation with a specific motor type using a value in the following parameter. NOTE: This command works in conjunction with the <i>Mode of Operation 'j'</i> command.	
Parameters:	Value	Parameter Description:
1	0 – F	A hex value from 0 to F indicating one of several possible motor selections for use with the Quick-Spin board. 0- AC Induction motor. 1- BLDC motor. Values 2 to F are reserved.
Returns:	'!'	ACK if successful.

Command:		Set "P" for Speed Gain PID. (DC motors only)
ASCII String:		'\$ P'
Parameters Required:		4 characters sent as a 16 bit hex value.
Total Tx Characters:		6
Description:	Sets the Proportional Gain variable of the DC motor Speed Gain PID algorithm.	
Parameters:	Value	Parameter Description:
1	0 – FFFF	A hex value from 0 to FFFF indicating Proportional gain of the Speed Gain PID.
Returns:	'!'	ACK if successful.

Command:		Set "I" for Speed Gain PID. (DC motors only)
ASCII String:		'\$ I'
Parameters Required:		4 characters sent as a 16 bit hex value.
Total Tx Characters:		6
Description:	Sets the Integral Gain variable of the DC motor Speed Gain PID algorithm.	
Parameters:	Value	Parameter Description:
1	0 – FFFF	A hex value from 0 to FFFF indicating Integral gain of the Speed Gain PID.
Returns:	'!'	ACK if successful.

Command:		Set "D" for Speed Gain PID. (DC motors only)
ASCII String:		'\$ D'
Parameters Required:		4 characters sent as a 16 bit hex value.
Total Tx Characters:		6

Description:	Sets the Differential Gain variable of the DC motor Speed Gain PID algorithm.	
Parameters:	Value	Parameter Description:
1	0 – FFFF	A hex value from 0 to FFFF indicating Differential gain of the Speed Gain PID.
Returns:	'!'	ACK if successful.

Command:	Set "P" for Current Gain PID. (DC motors only)	
ASCII String:	'\$ p'	
Parameters Required:	4 characters sent as a 16 bit hex value.	
Total Tx Characters:	6	
Description:	Sets the Proportional Gain variable of the DC motor Current Gain PID algorithm.	
Parameters:	Value	Parameter Description:
1	0 – FFFF	A hex value from 0 to FFFF indicating Proportional gain of the Current Gain PID.
Returns:	'!'	ACK if successful.

Command:	Set "I" for Current Gain PID. (DC motors only)	
ASCII String:	'\$ i'	
Parameters Required:	4 characters sent as a 16 bit hex value.	
Total Tx Characters:	6	
Description:	Sets the Integral Gain variable of the DC motor Current Gain PID algorithm.	
Parameters:	Value	Parameter Description:
1	0 – FFFF	A hex value from 0 to FFFF indicating Integral gain of the Current Gain PID.
Returns:	'!'	ACK if successful.

Command:	Set "D" for Current Gain PID. (DC motors only)	
ASCII String:	'\$ d'	
Parameters Required:	4 characters sent as a 16 bit hex value.	
Total Tx Characters:	6	
Description:	Sets the Differential Gain variable of the DC motor Current Gain PID algorithm.	
Parameters:	Value	Parameter Description:
1	0 – FFFF	A hex value from 0 to FFFF indicating Differential gain of the Current Gain PID.
Returns:	'!'	ACK if successful.

Command:	Get Set Point	
ASCII String:	'\$ A'	
Parameters Required:	None	
Total Tx Characters:	2	
Description:	Requests the 16 Speed/Direction Set Points. Set points can be preprogrammed and saved in non-volatile memory for later recall. Upon recall, 1 of the 16 non-volatile Set Points are loaded as the current setting. If disabled or in fault, the	

	drive does not become active through the use of this command. See the <i>Load Set Point 'e'</i> and <i>Save Set Point 'C'</i> commands for more info.	
Parameters:	Value	Parameter Description:
None		
Returns:		This command returns the following Set Point string.
83 characters as.. 2 prefix chars + 5x16 + 1 ACK char.	'#'	<ul style="list-style-type: none"> '#' Hash (Pound) indicates the start of a string sent from the Quick-Spin board that is something other than '!' (ACK).
	'P'	<ul style="list-style-type: none"> The 'P' indicates the following is a Set Point string.
	F or R	<p>16 sets of the following 6 characters starting with Set Point 0 and running to Set Point 15.</p> <ul style="list-style-type: none"> 1 character indicating motor direction in the form of: F – Forward motion. R – Reverse motion.
	0000 – FFFF	<ul style="list-style-type: none"> 4 characters as a 16 bit hex value indicating motor speed in RPM. NOTE: The speed value is subject to the minimum and maximum speed settings. Speed settings beyond the minimum and maximum settings are ignored and default to the min or max value.
	'!'	<ul style="list-style-type: none"> '!' ACK character.

Command:	Save Set Point	
ASCII String:	'\$ C'	
Parameters Required:	7 characters sent in various forms. See below.	
Total Tx Characters:	9	
Description:	Saves a direction and speed value to 1 of 16 set point locations in non-volatile memory. Set points can be preprogrammed and saved in non-volatile memory for later recall. Upon recall, 1 of the 16 non-volatile set points are loaded as the current motor speed and direction. Saved set point can also be default loaded on power up and/or while in operation through the use of a 4 bit input on the Quick-Spin board. Saving set points has no effect on the drive's current operation. See the <i>Load Set Point 'e'</i> , and <i>Get Set Point 'A'</i> commands for more info.	
Parameters:	Value	Parameter Description:
7	0 – F	<ul style="list-style-type: none"> 1 character as a hex nibble from 0 to F indicating 1 of 16 different motor speed and direction Set Points. <p>Values 0 to F (16) correspond to Set Point storage locations 0 to 15.</p>
	F or R	<ul style="list-style-type: none"> 1 character indicating motor direction in the form of: F – Forward motion. R – Reverse motion.
	0000 – FFFF	<ul style="list-style-type: none"> 4 characters as a 16 bit hex value indicating motor speed in RPM. NOTE: The speed value is subject to the minimum and maximum speed settings. Speed settings beyond the minimum

		and maximum settings are ignored and default to the min or max value.
Returns:	' ! '	ACK if successful.

Command:	Load Set Point	
ASCII String:	' \$ e '	
Parameters Required:	1 character sent as a hex nibble.	
Total Tx Characters:	3	
Description:	Loads a direction and speed value from 1 of 16 non-volatile set point locations as the motor's current speed and direction. Set points can be preprogrammed and saved in non-volatile memory for later recall. Saved set point can also be default loaded on power up and/or while in operation through the use of a 4 bit input on the Quick-Spin board. If the drive is clear of faults, loading a set point causes the motor to change to the new set point's speed and direction using the programmed acceleration and deceleration times. In cases where the new set point's speed value is beyond the programmed minimum or maximum speed values, the drive will stop at the minimum or maximum set value. Loading an unprogrammed Set Point results in a stopped condition. (zero speed, zero direction) rather than a minimum speed condition. See the <i>Save Set Point ' C '</i> , <i>Get Set Point ' A '</i> , <i>Set Acceleration Time ' T '</i> , <i>Set Deceleration Time ' V '</i> , <i>Set Minimum Speed ' L '</i> , and <i>Set Maximum Speed ' U '</i> commands for more info.	
Parameters:	Value	Parameter Description:
1	1 – F	1 character as a hex nibble from 1 to F indicating 1 of 16 different motor speed and direction Set Points.
Returns:	' ! '	ACK if successful.

5.4 Protocol B Details.

Protocol B is for use with terminal programs. It is a full, ASCII based, text menu system.

- Unused characters are ignored.
- At any time, Escape (*Esc*) may be pressed to abort a command or variable entry.
- Pressing Enter without a value keeps the current value of any parameter.
- Commands that require values to be entered also require Enter to be pressed. Other commands such as Forward, Reverse, etc. and Submenu commands do not require Enter.

5.4.1 Protocol B – Main Menu.

The following is the *Main Menu* for this protocol:

Quick-Spin Motor Control Menu	
(c) 2007 All American	
E	Emergency Stop (min. Decel time)
K	Coast to Stop (Kill PWM)
S	Controlled Stop with Decel
X	Stop with DC Injection Brake (AC only)
F	Forward at Set Speed
R	Reverse at Set Speed
N	Set New Speed (in RPM)
+	Increase Speed
-	Decrease Speed
Y	Status Info
s	Motor Parameters
Z	Reset Motor
?	Show this Menu

E - Emergency Stop (min. Decel time) – Stops the motor as quickly as possible.

K - Coast to stop (Kill PWM) – Stops the motor by stopping all drive signals to the board's drive module and allowing the motor to coast until stopped.

s - Controlled stop with Decel – Stops the motor by decelerating to 0 speed using the programmed Deceleration time.

X - Stop with DC Injection Brake (AC only) – Stops AC motors using a DC Injection Braking voltage which is set in the *Motor Parameters* sub menu.

The above *Stop* commands result in the following string being displayed:

Stopped using the 's' command.

Where 's' is one of the following strings:

- a. **Emergency Stop (min. Decel time)**
- b. **Coast to Stop (Kill PWM)**
- c. **Controlled Stop with Decel**
- d. **Stop with DC Injection Brake**

F - Forward at set speed – Initiates forward motion under the following conditions:

1. At stop, using the *Acceleration Time* set in the *Motor Parameters* submenu until reaching and maintaining the speed indicated by the *Set New Speed* command.

2. If a speed has not been set, the default speed is 0 RPM and the command is ignored producing no motion.
3. If selected while the drive is already producing forward motion, the command is ignored.
4. If selected while the drive is producing reverse motion, the drive decelerates using the *Deceleration Time* set in the *Motor Parameters* submenu until the *Minimum Speed* set in the *Motor Parameters* submenu is met. The drive will then reverse direction and accelerate as described in bullet 1.

R - Reverse at Set Speed – Initiates reverse motion under the following conditions:

1. At stop, using the *Acceleration Time* set in the *Motor Parameters* submenu until reaching and maintaining the speed indicated by the *Set New Speed* command.
2. If a speed has not been set, the default speed is 0 RPM and the command is ignored producing no motion.
3. If selected while the drive is already producing reverse motion, the command is ignored.
4. If selected while the drive is producing forward motion, the drive decelerates using the *Deceleration Time* set in the *Motor Parameters* submenu until the *Minimum Speed* set in the *Motor Parameters* submenu is met. The drive will then reverse direction and accelerate as described in bullet 1.

Both Forward and Reverse commands causes the following string to be displayed with the appropriate verbiage:

Motor ACCELERATING/DECELERATING in FORWARD/REVERSE motion to 'n' RPM.

Where 'n' is the current set speed in RPMs.

N - Set New Speed – Sets a new target speed for the connected motor. The system will display the following prompt for a new speed:

Current speed is 'n' RPM. Enter new RPM Speed:

Where 'n' is the current set speed.

'+' - Increase speed – Displays the following menu to allow the current motor

```
0: +1 RPM
1: +2 RPM
2: +5 RPM
3: +10 RPM
4: +25 RPM
5: +50 RPM
6: +100 RPM
7: +200 RPM
Enter Increment Amount:
```

speed to be increased by 1 of the following 8 possible RPM selections:

Upon completion, the command returns the following string:

Speed incremented by 'n' to 'x' RPM.

Where 'n' is the value of the increment step in RPM and 'x' is the motor's new set speed.

'-' - Decrease speed – Displays the following menu to allow the current motor speed to be decreased by 1 of the following 8 possible RPM selections:

```
0: -1 RPM
1: -2 RPM
2: -5 RPM
3: -10 RPM
4: -25 RPM
5: -50 RPM
6: -100 RPM
7: -200 RPM
Enter Decrement Amount:
```

Upon completion, the command returns the following string:

Speed decremented by 'n' to 'x' RPM.

Where 'n' is the value of the decrement step in RPM and 'x' is the motor's new set speed.

Y - status info – Display system information and variables as the following with the appropriate verbiage:

```

Quick-Spin System Status:

Motor Type: AC/BLDC
Drive Type: 's'
Current Motor direction: FORWARD/REVERSE/STOPPED
Current Set Motor Speed: 'speed in RPM' RPM
Current Set Motor Frequency: 'speed in hertz' Hertz
Actual Motor Speed: 'actual speed in RPM' RPM
Faults:
Over Voltage - OK/FAULT
Under Voltage - OK/FAULT
Over Current - OK/FAULT
Short Circuit - OK/FAULT
Over Temp - OK/FAULT
Hall Error - OK/FAULT
Motor Stall - OK/FAULT
Previous Faults:
Over Voltage - OK/FAULT
Under Voltage - OK/FAULT
Over Current - OK/FAULT
Short Circuit - OK/FAULT
Over Temp - OK/FAULT
Hall Error - OK/FAULT
Motor Stall - OK/FAULT
Motor Current: 'AA.aaa' Amps
DC Bus Voltage: 'VVV' Volts
Drive Temp: 'nnn' C
Ambient Temp: 'nnn' C
Speed Pot: '0-100%'

```

Where:

- 's' is one of the following drive types:
 - a. OPEN LOOP
 - b. SPEED PID W. 120 DEGREE HALL SENSORS
 - c. SPEED PID W. 60 DEGREE HALL SENSORS
 - d. SPEED AND CURRENT PID W. 120 DEGREE HALL SENSORS
 - e. SPEED AND CURRENT PID W. 60 DEGREE HALL SENSORS
 - f. ALL AMERICAN ONE SHUNT
 - g. RENESAS ONE SHUNT
- 'speed in RPM' is the current set speed in RPM.
- 'speed in hertz' is the current set speed in hertz.
- 'actual speed in RPM' is the actual speed in RPM based on input from the hall sensors.
- 'AA.aaa' is the DC buss current in amps and milliamps up to 99 amps and 900 milliamps with a 100mA resolution.
- 'VVV' is the buss voltage in voltage.
- 'nnn' is the drive and ambient board temperature in degrees Celsius.

- '0-100%' is a percentage representation of the board's speed control pot.

s - **Motor Parameters** – Displays the *Motor Parameters* submenu. See the *Protocol B – Motor Parameters Submenu* section for more info.

? - **show this Menu** – Displays (reprints) the *Main Menu*.

Unseen Command:

@@ - **Switch Protocols** – Pressing the '@' symbol twice causes the Quick-Spin board to switch to communication protocol A. Once switched, the Quick-Spin board accepts commands only in *Protocol A* format. Switching to *Protocol A* may be undesirable when using *Protocol B* since *Protocol A* is a terse (menu-less) ASCII protocol design for use with machine interfaces. See the *Protocol A Command Set* section for more info. Upon operation of this command, the following string is displayed:

WARNING: Switching to Protocol A will cause commands to change and all menus to disappear. Are you sure? y/N

If 'N' is selected the system returns to the *Main Menu*. Otherwise, the communication system is switched to *Protocol A* mode. No other actions are taken with the board. It remains in the same state. Once switched, the system must be reset to return to *Protocol B*.

5.4.2 Protocol B – Motor Parameters Submenu.

The following is the *Motor Parameters* submenu for Protocol B:

```

Quick-Spin Motor Parameters Menu
(c) 2007 All American

    j      Operation Mode
    k      Motor Type

    M      Set Number of Motor Poles
    U      Set Upper Speed Limit (Max Speed in RPM)
    L      Set Lower Speed Limit (Min Speed in RPM)
    T      Set Accel Time in Seconds (0-25.5 secs)
    V      Set Decel Time in Seconds (0-25.5 secs)
    B      Set Boost Voltage (0-50V)
    W      Set DC Injection Brake Voltage (AC only, 0-50V)
    b      Set Rated Voltage (0-499V)
    a      Set Rated Frequency (0-999Hz)
    G      Set Speed Gain Parameters (DC only)
    ?      Return to Main Menu
    s      Show this Menu

>>

```


j - **Operation Mode** – Selects the type of operating methodology by displaying the following:

The current Operation Mode is 's'.

Select a new Operation Mode:

- 0- Open loop (AC motors)
- 1- Speed PID w/ 120 degree hall sensors
- 2- Speed PID w/ 60 degree hall sensors
- 3- Speed and Current PID w/ 120 degree hall sensors
- 4- Speed and Current PID w/ 60 degree hall sensors
- 5- All American One Shunt
- 6- Renesas One Shunt

Selection:

Where 's' is the current *Operation Mode*.

Upon a valid selection, the board is configured internally to that methodology and displays the following string:

Mode 's' selected.

Where 's' is one of the *Operating Mode* listed above.

k - **Motor Type** – Selects the type of motor connected to the Quick-Spin board by displaying the following:

The current Motor Type is 's'.

Select a new Motor Type:

- 0- AC
- 1- BLDC

Selection:

Where 's' is the current *Motor Type*.

Upon a valid selection, the board is configured internally for the type of motor selected and displays the following string:

Motor 's' selected.

Where 's' is one of the *Motor Types* listed above.

M - **Set Number of Motor Poles** – Sets the number of motor poles by displaying the following string:

Current number of poles is 'n'. New pole count is [2,4,6, or 8]:

Where 'n' is the current *Number of Poles*.

U - Set Upper Speed Limit (Max Speed in RPM) – Sets the maximum speed for the attached motor. Upon selection of this command, the system will prompt the user with the following string:

Current Max Speed is 'n' RPM. New Max Speed in RPM is:

Where 'n' is the current *Upper Speed Limit*.

L - Set Lower Speed Limit (Min Speed in RPM) – Sets the minimum speed for the attached motor. Upon selection of this command, the system will prompt the user with the following string:

Current Min Speed is 'n' RPM. New Min Speed in RPM is:

Where 'n' is the current *Lower Speed Limit*.

T - Set Accel Time in Seconds (0-25.5 secs) – Sets the *Acceleration Time* in seconds from 0 to 25.5 seconds in 100mS increments. The following string is displayed:

Current Accel Time is 'nn.n' Seconds. New Accel time is (in tenths of seconds):

Where 'nn.n' is the current *Acceleration Time*.

V - Set Decel Time in Seconds (0-25.5 secs) – Sets the *Deceleration Time* in seconds from 0 to 25.5 seconds in 100mS increments. The following string is displayed:

Current Decel Time is 'nn.n' Seconds. New Decel time is:

Where 'nn.n' is the current *Deceleration Time*.

B - Set Boost Voltage – Sets the *Boost Voltage* by issuing the following string:

Current Boost Voltage is 'n' Volts. New Boost Voltage is (0-50V):

Where 'n' is the current *Boost Voltage*.

W - Set DC Injection Brake Voltage – Sets the *DC Injection Voltage* by issuing the following string:

Current DC Injection Voltage is 'n' Volts. New DC Injection Voltage is:

Where 'n' is the current *DC Injection Brake Voltage*.

b - Set Rated Voltage (0-499V) – Sets the *Rated Motor Voltage* of the current motor by issuing the following string:

```
Current Rated Voltage is 'n' Volts. New Rated Voltage is (0-499V):
```

Where 'n' is the current *Rated Motor Voltage*.

a - Set Rated Frequency (0-999Hz) – Sets the *Rated Motor Frequency* of the current motor by issuing the following string:

```
Current Rated Frequency is 'n' Hz. New Rated Frequency is (0-999Hz):
```

Where 'n' is the current *Rated Motor Frequency*.

G - Set Speed Gain Parameters – Displays the *Set Speed Gain Parameters* submenu. See the *Protocol B – Speed Gain Parameters Submenu* section for more info.

? - Return to Main Menu – Returns to the *Main Menu*. See the *Protocol B – Main Menu* section for more info.

s - show this Menu – Displays (reprints) the *Motor Parameter* submenu.

5.4.3 Protocol B – Speed Gain Parameters Submenu.

The following is the *Speed Gain Parameters* submenu for Protocol B:

```
Quick-Spin Speed Gain Parameters Menu
(c) 2007 All American
  P   Proportional Gain (0-99)
  I   Integral Gain (Not Implemented at this time)
  D   Differential Gain (Not Implemented at this time)

  s   Return to Motor Parameters
  G   Show this Menu
  ?   Return to Main Menu

>>
```

P - Proportional Gain (0-99) – Sets the *Proportional Gain* value for the *Speed Gain PID* algorithm by displaying the following string:

```
Current Proportional Gain is 'n'. New Proportional Gain is (0-99):
```

Where 'n' is the current *Proportional Gain* value for the *Speed Gain PID*.

I - Integral Gain – Sets the *Integral Gain* value for the *Speed Gain PID* algorithm by displaying the following string:

```
Current Integral Gain is 'n'. New Integral Gain is (0-99):
```

Where '*n*' is the current *Integral Gain* value for the *Speed Gain PID*.

D - Differential Gain – Sets the *Differential Gain* value for the *Speed Gain PID* algorithm by displaying the following string:

```
Current Differential Gain is 'n'. New Differential Gain is (0-99):
```

Where '*n*' is the current *Differential Gain* value for the *Speed Gain PID*.

s - Return to Motor Parameters – Returns to the *Motor Parameters*. See the *Protocol B – Motor Parameters Submenu* section for more info.

G - show this Menu – Displays (reprints) the *Speed Gain Parameters* submenu.

? - Return to Main Menu – Returns to the *Main Menu*. See the *Protocol B – Main Menu* section for more info.

5.4.4 Protocol B – Current Gain Parameters Submenu.

NOTE: THIS SECTION IS NOT IMPLEMENTED AT THIS TIME.

The following is the *Current Gain Parameters* submenu for Protocol B:

```
Quick-Spin Current Gain Parameters Menu
(c) 2007 All American

    p    Proportional Gain (0-99)
    i    Integral Gain (0-99)
    d    Differential Gain (0-99)

    s    Return to Motor Parameters
    H    Show this Menu
    ?    Return to Main Menu
..
```

p - Proportional Gain – Sets the *Proportional Gain* value for the *Current Gain PID* algorithm by displaying the following string:

```
Current Proportional Gain is 'n'. New Proportional Gain is (0-99):
```

Where '*n*' is the current *Proportional Gain* value for the *Current Gain PID*.

i - **Integral Gain** – Sets the *Integral Gain* value for the *Current Gain PID* algorithm by displaying the following string:

Current Integral Gain is 'n'. New Integral Gain is (0-99):

Where 'n' is the current *Integral Gain* value for the *Current Gain PID*.

d - **Differential Gain** – Sets the *Differential Gain* value for the *Current Gain PID* algorithm by displaying the following string:

Current Differential Gain is 'n'. New Differential Gain is (0-99):

Where 'n' is the current *Differential Gain* value for the *Current Gain PID*.

s - **Return to Motor Parameters** – Returns to the *Motor Parameters*. See the *Protocol B – Motor Parameters Submenu* section for more info.

H - **show this Menu** – Displays (reprints) the *Speed Gain Parameters* submenu.

? - **Return to Main Menu** – Returns to the *Main Menu*. See the *Protocol B – Main Menu* section for more info.

5.5 System Variables.

The following variables are either passed to and from the Quick-Spin board and GUI using Protocol A or modified via the Protocol B Menu system.

5.5.1 Values set by the GUI or Text Menu system.

Variable Name	Size	Description
SetSpeedInRpm	16 bit	Current Set Speed in RPMs.
AccelTime	8 bit	Time in 10ths of a second. So, max would be 25.5 seconds.
DecelTime	8 bit	Time in 10ths of a second. So, max would be 25.5 seconds.
UpperSpeedLimit	8 bit	Upper speed limit in hertz.
LowerSpeedLimit	8 bit	Lower speed limit in hertz.
BoostVoltage	8 bit	Boost voltage in volts.
MotorPoles	8 bit	Number of poles in a given motor.
InjBrakeVolts	8 bit	DC Injection brake voltage in volts.
GainParamP	16 bit	16 bit value for Gain PID parameter 'P'.
GainParamI	16 bit	16 bit value for Gain PID parameter 'I'.
GainParamD	16 bit	16 bit value for Gain PID parameter 'D'.
CurrentParamP	16 bit	16 bit value for Current PID parameter 'P'.
CurrentParamI	16 bit	16 bit value for Current PID parameter 'I'.
CurrentParamD	16 bit	16 bit value for Current PID parameter 'D'.
MotorType	8 bit	Contains flags for the type of motor used.
OpMode	8 bit	Contains flags for the type of operating mode.
RatedFreq	16 bit	Nameplate rating of the motor's max freq.

RatedVolts	16 bit	Nameplate rating of the motor's max volts.
SetPoint[16]		Array of 16 Set Points. Each Set Point consists of the following:
Direction	8 bit	Motor Direction for the Set Point.
SpeedInRpm	16 bit	Speed setting for the Set Point.

5.5.2 Values created by the MCU sent to the GUI or Text Menu system.

Variable Name	Size	Description
ActualSpeedInRpm	16 bit	The current speed in RPM. Note, this is not the set speed.
CurFaults	8 bit	Flags to indicates current fault status as follows:
Bit 0: Over Voltage		1- Fault, 0- Ok.
Bit 1: Under Voltage		
Bit 2: Over Current		
Bit 3: Short Circuit		
Bit 4: Over Temperature		
Bit 5: Hall Sensor Error		
Bit 6: Motor Stall		
Bit 7: Emergency Stop Loop		1- Loop Open, 0- Loop Closed.
LastFaults	8 bit	Flags to indicates previous fault status.
SystemFlags	8 bit	Flags to indicates system status as follows:
Bit 0: Direction, bit 0		00- Motor off, 01- Motor spinning forward,
Bit 1: Direction, bit 1		10-Motor spinning in reverse, 11- Reserved.
Bit 2: DriveEnable		1- Drive enabled, 0- Drive disabled.
Bit 3: HallSensor1		1- Hall output, 0- No hall output.
Bit 4: HallSensor2		1- Hall output, 0- No hall output.
Bit 5: HallSensor3		1- Hall output, 0- No hall output.
Bit 6: PendingDirection, bit 0		Same as direction, but a pending command.
Bit 7: PendingDirection, bit 1		
BusVoltage	16 bit	Bus voltage in voltage.
BusCurrent	16 bit	Bus current in milliamps.
DriveTemp	8 bit	Drive temperature in degrees C.
AmbientTemp	8 bit	Ambient temperature in degrees C.
PotPosition	8 bit	Value of the board's pot from 0-100%
PhaseInfo	8 bit	Flags indicating PWM relationships and phase as follows:
Bits 0 and 1 as drive leg U		00 – Leg off. (Not Driving.)
Bits 2 and 3 as drive leg V		01 – Leg is driving and is the 0° leg.
Bits 4 and 5 as drive leg W		10 – Leg is driving and is the 60° leg.
Bits 6 and 7 are reserved		11 – Leg is driving and is the 120° leg.

6. GUI Interface.

6.1 GUI Compatibility.

The PC based GUI is designed for the Microsoft Windows 2000 or Windows XP operating system. It is written in C++ using Microsoft Visual Studio 6.

- Optionally, an Eclipse plug-in GUI may be provided at a future date.
- Optionally, a Renesas HEW plug-in GUI may be provided at a future date.

6.2 GUI Structure.

The GUI shall be written and maintained such that it provides a clean and conscience methodology for being expanded, added to, subtracted from, or otherwise changed.

To assist in the separation of functions and make the GUI code more useable and portable, 5 software and 1 hardware layers have been defined which loosely correspond to the standard 7 layer OSI Model. When writing software code, every effort shall be made to maintain separate between layers including the use of separate .c, .cpp, .h, and .hpp files. The layers are defined as follows:

- **GUI Controls – OSI Layer 7, Application Layer.** Code in this layer handles the GUI controls and Windows interface. This code is dependant on the operating system, build, and application environment. Ie: A GUI built as a Dialog Box in Windows using Visual Studio 6 vs. a GUI built for the Renesas HEW tool.
- **GUI Interface – OSI Layer 6, Presentation Layer.** Code in this layer works to transform and interpret information between the Application Layer and the Session Layer. Essentially, this layer is a series of wrapper functions that hide structural details from the Command Processor.
- **Command Processor – OSI Layer 5, Session Layer.** This layer handles all of the actual data processing and functionality done in the GUI system. It communicates to the user upwards through OSI layers 6 and 7 and communicates with the Quick-Spin board downwards through OSI layers 4 and 2.
- **Comm. Interface – OSI Layer 4, Transport Layer.** This layer is essentially another series of wrapper functions allowing the Command Processor to communicate with know functions while hiding their actual implementation.
- **Comm. I/O – OSI Layer 2, Data Link Layer.** This is the lowest software layer of the OSI model. The code in this layer handles communication and basic protocol details. Basically, a communication driver. For example, code in this layer would be differ from a serial vs. a USB implementation.
- **Hardware – OSI Layer 1, Physical.** This is the only layer in the OSI model which is not a software layer, but rather a hardware layer. For the Quick-Spin GUI application, this layer is inherently designed into the hardware of the Quick-Spin board and Host PC.

- **OSI Layer 3, Network.** This OSI layer handles routing and switching operations between nodes. It is currently not implemented in the Quick-Spin board or GUI as the system is currently designed for single board/single GUI operation. However, changes to the system to incorporate multi node protocols such as RS-422 or RS-485 could take advantage of the addition of this layer.

6.3 GUI Overview.

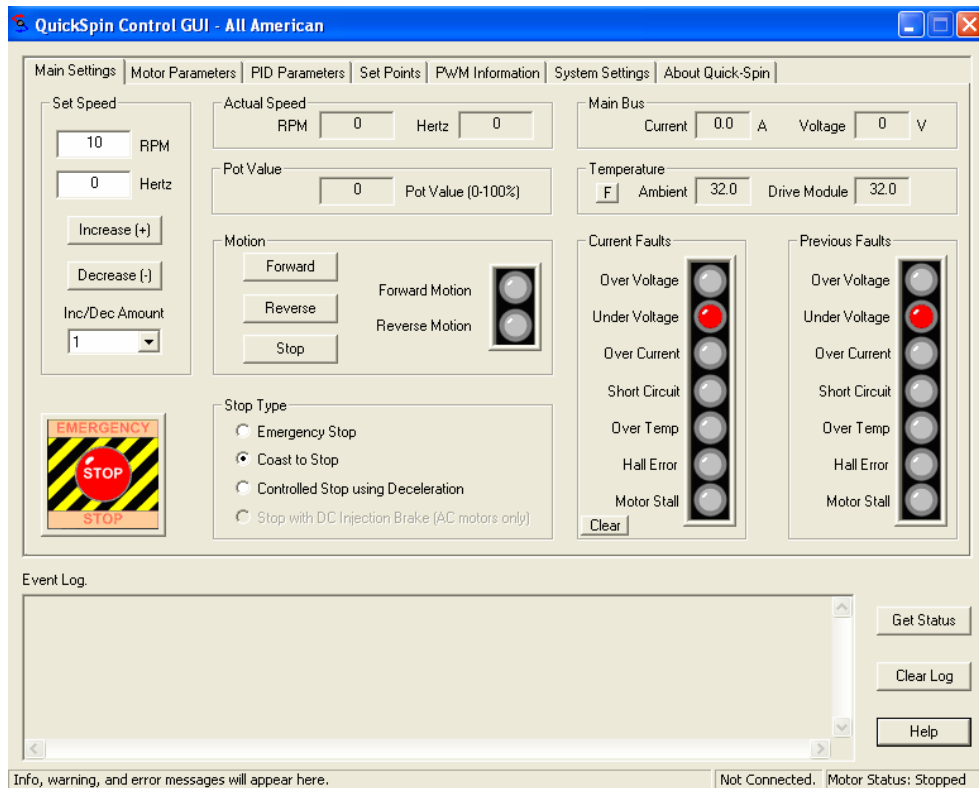
The following is a list of features and attributes that shall be incorporated into and/or inherent of, the All American Quick-Spin control GUI regardless of it's actual form, appearance, or implementation.

- The Quick-Spin GUI shall be design to allow a user to interface to, and manipulate data to and from the Quick-Spin board using communication Protocol A.
- The GUI shall be intuitive and user friendly requiring little to no skill for operation other than basic Windows skills.
- The GUI shall provide more control and substance than simple, "eye candy" GUIs as seen in some competitor products. Further, the GUI shall be void of inane or overly complex settings also found on the GUIs of other competitor products.
- Dangerous or consequential settings in the GUI system shall always result in prompting from the GUI to make sure the user wants to make changes.
- All buttons, readouts, and other controls shall be help context sensitive allowing the user to call up a help system at any time for information.
- Optionally, the GUI shall contain a script engine and processor allowing users to build, save, open, and execute command scripts. When executing, command scripts shall have the ability to alter settings in a timed fashion thus providing a method of automation to the GUI useful for demos, tests, and other repeatable tasks.

6.4 GUI Implementation in Windows 2000/XP.

This section details the various windows and controls in the Quick-Spin GUI.

The GUI shall be a non sizeable Dialog Box utilizing "tabbed" pages to access various controls as shown. Each tab shall contain an "Emergency Stop" control for quickly stopping the system.



The lower face of the GUI consists of an Event Log which records messages from the GUI including commands, errors, warnings, and other information as well as data received from the Quick-Spin system.

Also found on the lower face of the GUI are buttons to manually retrieve the Quick-Spin system's status, clear the Event Log window, and obtain GUI help.

The bottom of the GUI contains a status bar which prints current message from the GUI and indicates connection and motor status.

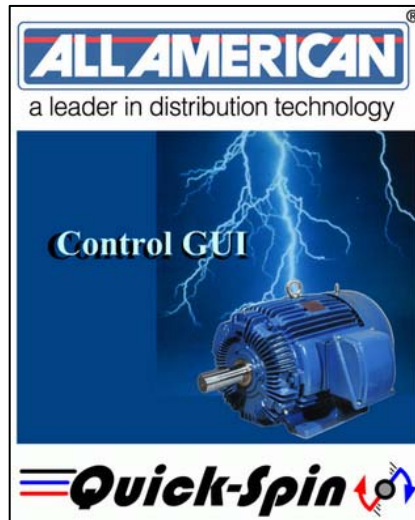
6.4.1 GUI Configuration File.

The GUI Program uses a proprietary, text based configuration file called '*QuickSpin.gcf*'. This file saves all pertinent motor parameters when the GUI is closed. Upon GUI startup, an option is available to recall these settings.

The configuration file can be deleted at any time without harm. If GUI is started, and the configuration file is not found, default parameters within the GUI can be used.

6.4.2 Splash Screen.

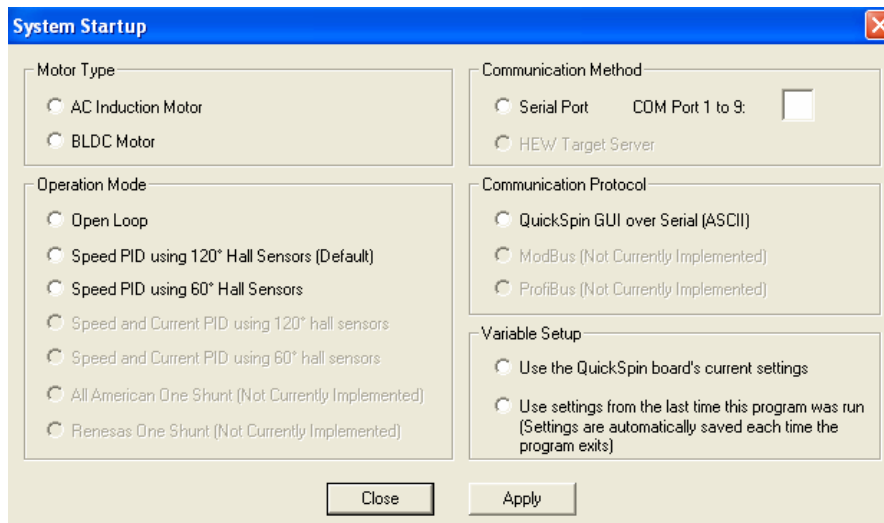
When the Quick-Spin GUI is started, a product splash screen appears for a few seconds along with the sound effect of a starting motor.



At any time during this splash screen, the 'ESC' can be pressed to bypass the splash screen and sound effect.

6.4.3 System Startup Dialog Box.

When the GUI is started, the 'System Startup Dialog Box' appears. This box allows for several critical configurations. At any time, the 'ESC' button can be pressed to exit.



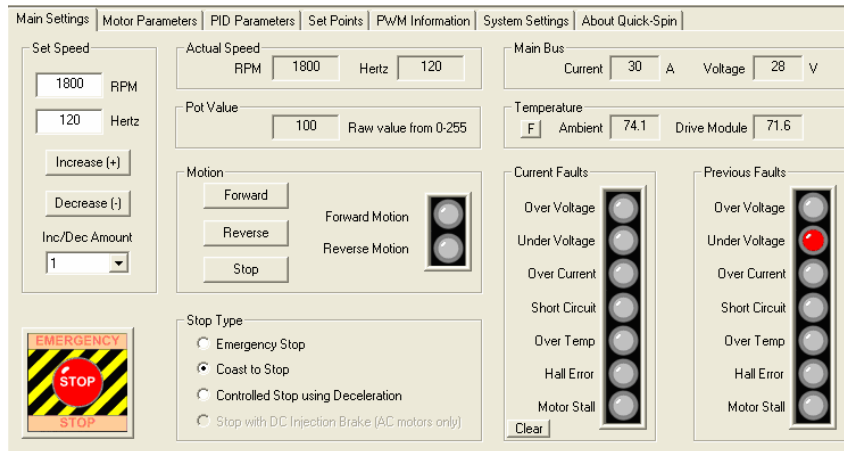
- **Motor Type.** Allows the user to select drive configuration for either an AC Induction or BLDC (Brushless DC) motor.

- **Operation Mode.** Allows the selection of several types of possible drive schemes for the attached motor. Note, some of these schemes are not implemented.
- **Communication Method.** There are two options:
 - **Serial Port.** This method uses a standard serial port from COM 1 to 9. Virtual serial ports over USB are allowed.
 - **HEW Target Server.** This option works in conjunction with the Renesas HEW Target Server. The physical link is the Renesas E8 operating over USB and connected to the Quick-Spin's E8 Debug port. NOTE, this method is not currently implemented.
- **Communication Protocol.** Default selection is the Quick-Spin's 'Protocol A' GUI protocol. Options for the industry standard, ModBus and ProfiBus protocols are present, however, these protocols have not been implemented.
- **Variable Setup.** There are two options:
 - **Use the Quick-Spin board's current settings.** This selection asks the Quick-Spin board for its current configuration and loads this data into all GUI variables.
 - **Use Settings from the Last Time the Program was Run.** This selection retrieves motor parameters from the GUI Configuration File (QuickSpin.GCF) and loads them into the GUI's variables. If the configuration file cannot be found, default variables within the GUI are used.
- **Apply Button.** This button applies the current selections on the 'System Startup Dialog Box'. If needed, the GUI communicates with the Quick-Spin system to retrieve or send parameters. At any time, selections can be changed and the 'Apply' button clicked again. For example, if 'Apply' is clicked to retrieve board parameters, selections on the 'System Startup Dialog Box' may change. If undesirable, these selections can be modified and 'Apply' can be used to send the new selections back to the Quick-Spin system.
- **Close Button.** Clicking the 'Close' button will close the 'System Startup Dialog Box' and take you into the GUI program. Note, you must click the 'Apply' button after making your selections. Clicking 'Close' without clicking 'Apply' will abandon your changes. If this is the case, you will be asked if this is what you want to do. If the program cannot continue based

on your selections, the GUI will exit, otherwise you will enter the GUI program.

6.4.4 Main Settings Tab.

The first tab of the GUI, *'Main Settings'* contains controls for speed, stop types, forward and reverse motion, fault indicators, and a range of status fields.



- **Set Speed.** Enter the desired speed of the motor here in either RPMs or hertz. Entering a speed value does not start motion, but will dynamically change the motor's speed if already running. Use the *'Forward'* and *'Reverse'* button to initiate motion. Note the values here are subject to the Minimum and Maximum Speed values found on the *'Motor Parameters'* tab.
- **Increase (+) and Decrease (-) Speed Buttons.** These buttons add or subtract RPMs to the current Set Speed value. The amount added or subtracted is set in the *'Inc/Dec Amount'*.
- **Inc/Dec Amount.** Selects 1 of 8 different amounts to increment or decrement the current Set Speed by. This value is used the *'Increase (+)'* and *'Decrease (-)'* buttons.
- **Actual Speed.** This is the actual motor speed in RPMs and hertz. This value is determined by hall sensor data and may be different that the Set Speed.
- **Pot Value.** This is a percentage, from 0 to 100%, of the Quick-Spin system's speed pot position.
- **Forward Button.** Click to start forward motion at the Set Speed.

- **Reverse Button.** Click to start reverse motion at the Set Speed.

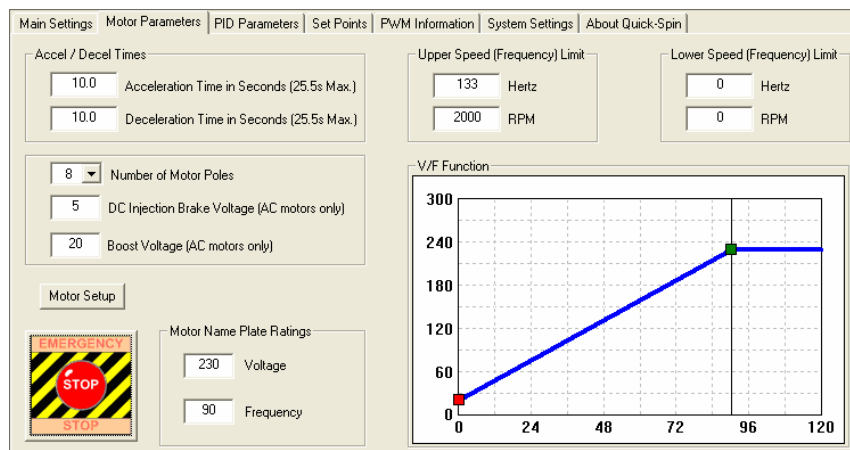
Note, if the motor is running in the opposite direction, a Stop command is automatically issued before changing directions.

- **Forward and Reverse Motion LEDs.** These two LEDs indicate either forward or reverse motion. A blinking LED indicates a command has been given for motion in the indicated direction, but the motor is not currently traveling in that direction.
- **Stop Button.** Stops the motor using one of the selected '*Stop Types*'.
- **Stop Type.** Selection of 4 different stop types used when the '*Stop*' button is pressed.
 - **Emergency Stop.** Controlled stop using the quickest time possible.
 - **Coast to Stop.** Shuts down all motor drive and allows the motor to mechanically coast to a stop.
 - **Controlled Stop using Deceleration.** Controlled stop using the current '*Decel Time*' value found on the '*Motor Parameters*' tab.
 - **Stop with DC Injection Brake (AC Motor Only).** Controlled stop for AC motors only by which a small DC snubber voltage is injected into the motor. See the settings on the '*Motor Parameters*' tab.
- **Main Bus Current.** Displays the approximate current of the main bus.
- **Main Bus Voltage.** Displays the approximate voltage of the main bus.
- **Temperature.** There are two different temperature values. Both values can be displayed in Fahrenheit or Celsius by clicking on the small 'F' or 'C' button.
 - **Ambient.** This displays the ambient temperature around the Quick-Spin system. The sensor is located near the lower left corner of the main board.
 - **Drive Module.** This displays the drive module's temperature. The sensor is located in the fins of the module's heatsink.
- **Current Faults and Previous Faults.** There are two identical Fault blocks, one for current faults, and one for old faults. The fault LEDs are as follows:

- **Over Voltage.** Indicates a bus over voltage condition. I.e: The main bus's voltage is too high for the given motor or system design.
- **Under Voltage.** Indicates a bus under voltage condition. I.e: the motor is drawing too much voltage for the given input voltage or current.
- **Over Current.** Indicates the motor is drawing too much current from the main bus.
- **Short Circuit.** Indicates a short circuit exists between 1 or more of the 3 motor drive legs or between 1 or more of the 3 drive leg and ground.
- **Over Temp.** Indicates the drive module is at it's rated temperature and must be shut down to prevent damage.
- **Hall Error.** Indicates a hall sensor error. This could be a faulty hall device, a bad connection, incorrect motor movement, or other such conditions.
- **Motor Stall.** Indicates the motor has stalled. This could be from the motor being mechanically restricted from starting, mechanically forced to stop while running, or being undersized for a particular purpose.
- **Fault Clear Button.** Click this button once to clear current faults and twice to clear both current and old faults.

6.4.5 Motor Parameters Tab.

The second tab of the GUI, '*Motor Parameters*' contains controls to setup various default and limit parameters for a given motor.

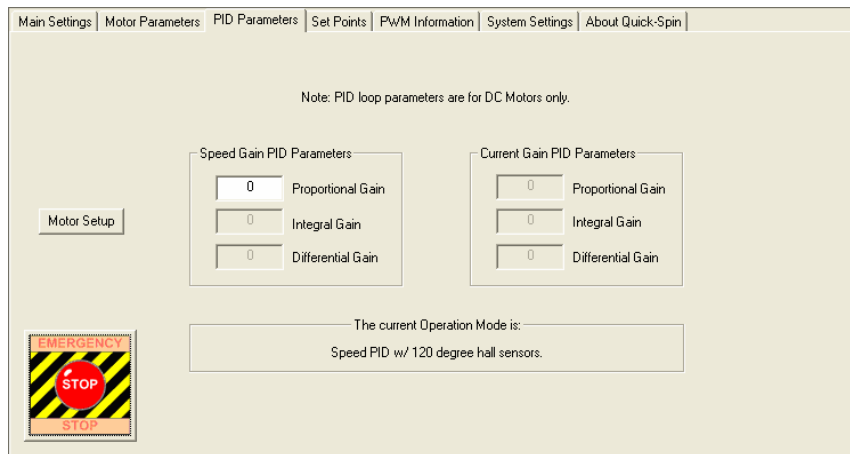


- **Accel / Decel Times.** These variables set the motor's acceleration and deceleration times. Values can be from 0 to 25.5 seconds in 10th second increments.
- **Number of Motor Poles.** This is the number of poles in the connected motor. This value can be found on the motor's nameplate or spec sheet. Default is 8 poles.
- **DC Injection Break Voltage (AC motors only).** This is the amount of DC voltage that is added to an AC motor to snub (stop) the motor. See the '*Stop with DC Injection Brake (AC Motor Only)*' on the '*Main Settings*' tab for more info.
- **Boost Voltage (AC motors only).** This is the boost voltage for an AC motor's V/F curve. This value influences the '*V/F Function*' graph.
- **Motor Setup Button.** This button opens the '*System Startup Dialog Box*' to allow the adjustment of additional, critical, motor parameters. See '*System Startup Dialog Box*' for more info.
- **Motor Name Plate Ratings, Voltage and Current.** These two values are motor parameters that can be found on the motor's nameplate or spec sheet. While used for both AC and DC motors, these parameters influence the '*V/F Function*' graph for AC motors only.
- **Upper Speed (Frequency) Limit.** This parameter sets the motor's upper speed limit and can be specified in RPMs or hertz. If a value is entered which is less than the current Set Speed, the current Set Speed is reduced to match. See '*Set Speed*' on the '*Main Settings*' tab for more info.
- **Lower Speed (Frequency) Limit.** This parameter sets the motor's lower speed limit and can be specified in RPMs or hertz. If a value is entered which is more than the current Set Speed, the current Set Speed is increased to match. See '*Set Speed*' on the '*Main Settings*' tab for more info.
- **V/F Function Graph (AC motors only).** This graph control allows the user to graphically set an AC motor's control curve. This control is grayed out (disabled) when using BLDC motors. There are two points on the graph covering 3 parameters:
 - **Red Point.** This point can only move in a Y direction and sets an AC motor's Boost Voltage. The actual setting will appear if the mouse is hovered over the point.

- **Greed Point.** This point can be moved in both the X and Y directions. The Y direction sets an AC motor's Rated Voltage while the X direction sets an AC motor's Rated Frequency. The actual setting will appear if the mouse is hovered over the point.

6.4.6 PID Parameters Tab.

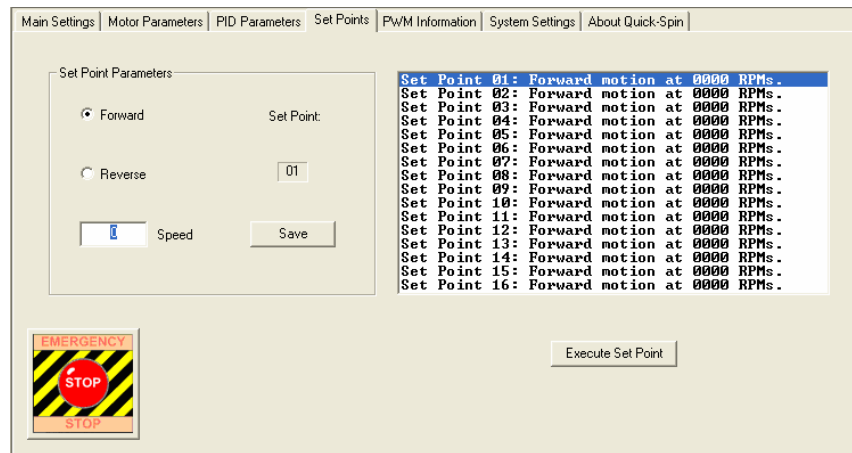
The third tab, '*PID Parameters*' contains controls related to the PID Operation Modes used for BLDC motors. When using an AC motor scheme, these controls are grayed out (disabled). Currently, only a Speed PID operation mode is implemented and within that algorithm, only the '*P*' parameter is currently adjustable. Therefore, many of this tab's controls are grayed out (disabled). See the '*System Startup Dialog Box*' section for more information on '*Operation Modes*'.



- **Motor Setup Button.** This button opens the '*System Startup Dialog Box*' to allow the adjustment of critical motor parameters and '*Operation Modes*'. See '*System Startup Dialog Box*' for more info.
- **Speed Gain PID Parameters.** These 3 values control the *Proportional (P)*, *Integral (I)*, and *Differential (D)*, values for a Speed based PID algorithm. In the current case, only the '*P*' parameter of All American's Speed PID is adjustable. Therefore, the '*I*' and '*D*' are grayed out (disabled).
- **Current Gain PID Parameters.** These 3 values are similar to the '*Speed Gain PID Parameters*'. However, a *Current Gain PID* algorithm is not currently implemented, therefore these controls are grayed out (disabled).
- **The Current Operation Mode is.** This control shows what '*Operation Mode*' (PID or not) is currently being used by the system. This values changes based on selections in the '*System Startup Dialog Box*'. See that section for more info.

6.4.7 Set Points Tab.

The fourth GUI tab, 'Set Points' contains controls to manipulate 16 different Set Points stored in the Quick-Spin system. While these Set Points are currently stored in volatile memory, embedded code could be modified to store such values in onchip Data Flash. Each Set Point defaults to 0 RPMs and Forward direction. Set Points are also stored in the GUI's configuration file and their values can be reloaded when a GUI session is started. See 'GUI Configuration File' for more info.



All 16 Set Points can be seen in the right hand list along with that Set Point's programmed Direction and Speed. Clicking on a Set Point in the list loads its current values into the controls on the left hand side. These parameters can be adjusted and saved back into the Set Point by clicking on the 'Save' button.

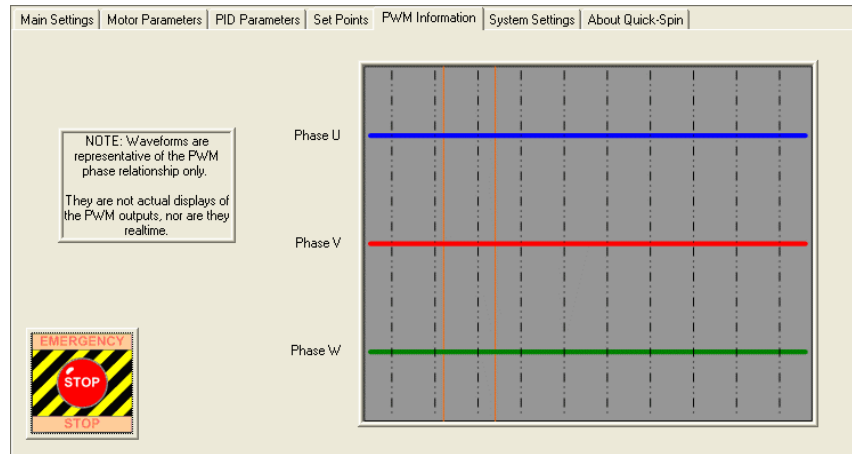
At any time, a Set Point can also be selected in the list and the 'Execute Set Point' button can be clicked to run the Set Point (I.e: Make the motor's current Direction and Speed the same as the Set Point.)

Note, when executing Set Points, only values appearing in the list are valid. I.e: A Set Point must be saved before it can be executed otherwise, the Set Point's original values are executed.

6.4.8 PWM Information Tab.

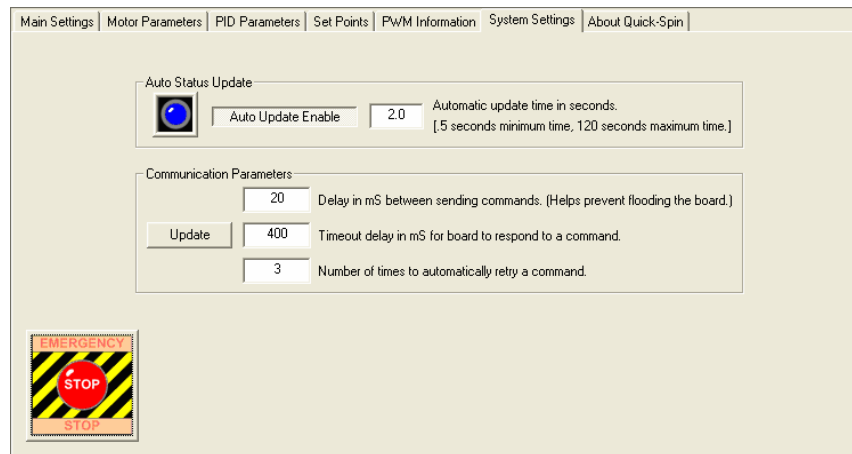
The fifth GUI tab, 'PWM Information' simply displays a graphical representation of data from the Hall sensors on a BLDC motor. The graph will remain flat when Hall sensors are not used. I.e, AC motors.

The information on the graph is for reference only and shows phase relationship between the sensors. The waveforms are not actual display outputs from the PWM drive, not are they realtime.



6.4.9 System Settings Tab.

The sixth tab, 'System Settings' contains controls related to the operation of the GUI itself.



- Auto Status Update.** When enabled via the 'Auto Update Enable' button, the GUI automatically sends a 'Status' command to the Quick-Spin system which results in all the GUI parameters being updated. The interval at which the 'Status' command is automatically sent is set in the left hand box and can be anything from .5 seconds to 120 seconds in 10ths of a second intervals. While the 'Auto Status Update' is operating, the blue indicator LED blinks at the programmed interval. To disable the 'Auto Status Update' click on the 'Auto Update Enable' button. It's text will

change to *'Auto Update Disabled'* and the LED will stop blinking. The default time is 2.0 seconds.

- **Communication Parameters.** This section allows 3 different parameters used in serial communications to be adjusted. These parameters can be changed at random, but the *'Update'* button must be clicked to actual load the new values into the communication subsystem. The values are as follows:
 - **Delay in mS between sending commands. (Flood control).** This value allows the user to slow down how fast a string of consecutive commands can be sent to the Quick-Spin system. Default is 20mS.
 - **Timeout delay in mS for board to respond to a command.** This value controls how long the GUI should wait before it considers the lack of response from a Quick-Spin system a problem. If the Quick-Spin system is programmed to do a lot of work, or if the user is using a USB to Serial converter, this parameter may need adjusting. Default time is 400mS.
 - **Number of times to automatically retry a command.** When the GUI sends a command to the Quick-Spin system and the command fails, (Ie: The Quick-Spin system does not return the appropriate response and in a timely manner.) the GUI will automatically try the command again. The value here tells the GUI how many times to retry a command before it fails for good. The default is 3 times.

6.4.10 About Quick-Spin Tab.

The final tab, *'About Quick-Spin'* is a credits and hyperlink page. This tab contains credits for the Quick-Spin system and project. It also contains the logos of the 3 companies involved in the project. Each log is hyperlinked, so clicking on a logo takes you to that company's website.

Music starts playing when this tab is opened. To stop the music, press the *'ESC'* button at any time.



6.5 Non Windows GUI Implementation.

There are currently no other implementations for the Quick-Spin GUI. However, the GUI's full Visual C++ source code is available. Further, the primary C++ object which handles all communication with the Quick-Spin system is created as a pure C++ object rather than a Windows reliant object.

7. Windows GUI Code.

7.1 Introduction.

To aide the developer in getting started, the entire Windows GUI code is available. The code is written in Visual Studio 6.0 C++ and uses MFC to create and interact with Windows controls.

Several efforts were taken to aide in the development of future code and to utilize the Quick-Spin GUI as a jumping off point for development. Some of those efforts include:

- Commenting code as much as possible.
- Helper functions and routines have been written, whether used or not.
- Wrapper objects to isolate MFC, non MFC, and pure C++ objects.
- Object designed with O/S portability in mind.
- Blank space left in GUI tabs for addition functions.
- Available space for additional tabs.
- Hooks for other means of communication.
- Hooks for other Operation Modes.
- Hooks for PID control.
- Hooks and function wrappers for HEW Target Server.
- Command line interface.
- Debug file that records all Log events with timestamps. (command line option.)

7.2 Non Windows, C++ Motor Control Object.

The primary class object used to control and communicate the Quick-Spin hardware system is called CMotor. This object is a pure C++ class and does not rely on MFC or Windows code allowing it to be ported to other operating systems such as Linux or an embedded system.

Since system features of a typical O/S are still required by CMotor, such as timers, and event handlers, a wrapper class, CMotorMFC, is also implemented as a means to virtualize the CMotor object.

7.3 Files.

The following is a list of the code files for the Quick-Spin GUI. In addition, various other Visual Studio 6.0 files also exist. However, the complete project is available. The project also contains an Open Source, serial communications library which is not listed here.

File Name	Purpose
Motor Class Files.	Files related to the primary Motor class.
Motor.cpp / .h	Contains the primary C++ class to control Quick-Spin hardware.
MotorMFC.cpp / .h	Contains wrapper class for CMotor.
MotorInfo.h	Contains parameter structs and info related to the CMotor class.
Tab Files	Files related to the GUI tabs.
TabAbout.cpp / .h	Code to handle the About Quick-Spin tab.
TabCtrlPlus.cpp / .h	Code to handle the switching and creation of all tabs.
TabMain.cpp / .h	Code to handle the Main Settings tab and its controls.
TabMotorParam.cpp / .h	Code to handle the Motor Parameters tab and its controls.
TabPID.cpp / .h	Code to handle the PID Parameters tag and its controls.
TabPWM.cpp / .h	Code to handle the PWM Information tab.
TabSetPoints.cpp / .h	Code to handle the Set Points tab and its controls.
TabSystemSetup.cpp / .h	Code to handle the System Settings tab and its controls.
INI Class Files.	Files related to the GUI Configuration File.
iniFile.cpp / .h	Code to create, read, write, and manipulate INI files.
Custom Control Files.	Files related to custom control created for this GUI.
CreditsThread.cpp / .h	Thread handler code for a scrolling credits thread.
GDIThread.cpp / .h	Code to operate and control the credits thread.
EStopBtn.cpp / .h	Code to create and handle a custom E-Stop button.
InputDialog.cpp / .h	Code to create and handle a custom Input Dialog Box.
MessageDlg.cpp / .h	Code for a custom, modal dialog message box.
Splash.cpp / .h	Code for the custom, startup dialog and sound effect.
StartupDialog.cpp / .h	Code for the System Startup Dialog Box.
VFControl.cpp / .h	Code for the custom V/F Function graph.
System Files.	Files related to the overall GUI.
ComWrapper.cpp / .h	Wrapper class for the communication functions.
QuickSpin.cpp / .h	Main application Windows startup code.
QuickSpinDlg.cpp / .h	Main application and handling code.

Defines.h	Various #defines and system constants.
MemDC.h	Defines a MemDC class for custom V/F Function control.
SerialMFC.h	Header file for the serial communication library.

8. Troubleshooting.

Issue.	Possible Solution.
User I/O and Main board won't communicate.	<ul style="list-style-type: none"> • Bad software – Test w/ default software. • Reset both boards. • Ensure User I/O board is correctly fitted to Main board.
GUI won't connect.	<ul style="list-style-type: none"> • Check cables. • Ensure correct COM port. • Check BAUD rate. • If USB to Serial Converter, try dedicated serial port. • Reset boards. • Ensure all other programs requiring the same COM port are closed. • Verify system by attempting to communicate using Text Menu Protocol and HyperTerminal.
Motor won't spin.	<ul style="list-style-type: none"> • Verify motor drive legs and hall sensors lines are correctly wired. Swap phases to check and test. • Verify correct motor parameters. • Verify correct Operation Mode and Motor Type settings. • Verify hall sensors operate at +5VDC only. • Verify correct input voltage and current to Quick-Spin. • Verify communication. Try using manual buttons. • Verify Speed Pot is not full CCW. • Verify Gate Voltage LED is lit. • Test with default code.

End of the Quick-Spin User Manual.

