



# **AX2550 AX2850**

## **Dual Channel High Power Digital Motor Controller**

### **User's Manual**



v1.9b, June 1, 2007

visit [www.roboteq.com](http://www.roboteq.com) to download the latest revision of this manual

©Copyright 2003-2007 Roboteq, Inc.

---



---

**Revision History**

<b>Date</b>	<b>Version</b>	<b>Changes</b>
June 1, 2007	1.9b	Added Output C active when Motors On Fixed Encoder Limit Switches Protection in case of Encoder failure in Closed Loop Speed Added Short Circuit Protection (with supporting hardware) Added Analog 3 and 4 Inputs (with supporting hardware) Added Operating Mode Change on-the-fly Changeable PWM frequency Selectable polarity for Dead Man Switch Modified Flashing Pattern Separate PID Gains for Ch1 and C2, changeable on-the-fly Miscellaneous additions and correction Added Amps Calibration option
January 10, 2007	1.9	Changed Amps Limit Algorithm Miscellaneous additions and correction Console Mode in Roborun
March 7, 2005	1.7b	Updated Encoder section.
February 1, 2005	1.7	Added Position mode support with Optical Encoder Miscellaneous additions and corrections
April 17, 2004	1.6	Added Optical Encoder support
March 15, 2004	1.5	Added finer Amps limit settings Enhanced Roborun utility
August 25, 2003	1.3	Added Closed Loop Speed mode Added Data Logging support Removed RC monitoring
August 15, 2003	1.2	Modified to cover AX2550 controller design Changed Power Connection section
April 15, 2003	1.1	Added analog mode section Added position mode section Added RCRC monitoring feature Updated Roborun utility section Modified RS232 watchdog
March 15, 2003	1.0	Initial Release

The information contained in this manual is believed to be accurate and reliable. However, it may contain errors that were not noticed at time of publication. User's are expected to perform their own product validation and not rely solely on data contained in this manual.



	Revision History	3
<b>SECTION 1</b>	Important Safety Warnings	13
	This product is intended for use with rechargeable batteries	13
	Do not Connect to a RC Radio with a Battery Attached	13
	Beware of Motor Runaway in Improperly Closed Loop	13
<b>SECTION 2</b>	AX2550 Quick Start	15
	What you will need	15
	Locating the Switches, Wires and Connectors	15
	Connecting to the Batteries and Motors	17
	Connecting to the 15-pin Connector	18
	Connecting the R/C Radio	19
	Powering On the Controller	20
	Button Operation	20
	Default Controller Configuration	21
	Connecting the controller to your PC using Roborun	22
	Obtaining the Controller's Software Revision Number	23
	Exploring further	24
<b>SECTION 3</b>	AX2550 Motor Controller Overview	25
	Product Description	25
	Technical features	26
<b>SECTION 4</b>	Connecting Power and Motors to the Controller	29
	Power Connections	29
	Controller Power	30
	Controller Powering Schemes	32
	Powering the Controller from a single Battery	32
	Powering the Controller Using a Main and Backup Battery	33
	Connecting the Motors	34
	Single Channel Operation	35
	Converting the AX2550 to Single Channel	35
	Power Fuses	36
	Wire Length Limits	37
	Electrical Noise Reduction Techniques	37
	Power Regeneration Considerations	37
	Overvoltage Protection	38
	Undervoltage Protection	38
	Using the Controller with a Power Supply	39

<b>SECTION 5</b>	General Operation	<b>41</b>
	Basic Operation	<b>41</b>
	Input Command Modes	<b>41</b>
	Selecting the Motor Control Modes	<b>42</b>
	Open Loop, Separate Speed Control	<b>42</b>
	Open Loop, Mixed Speed Control	<b>42</b>
	Closed Loop Speed Control	<b>43</b>
	Close Loop Position Control	<b>43</b>
	User Selected Current Limit Settings	<b>44</b>
	Temperature-Based Current Limitation	<b>44</b>
	Battery Current vs. Motor Current	<b>45</b>
	Regeneration Current Limiting	<b>46</b>
	Programmable Acceleration	<b>47</b>
	Command Control Curves	<b>48</b>
	Left / Right Tuning Adjustment	<b>49</b>
	Activating Brake Release or Separate Motor Excitation	<b>51</b>
	Emergency Shut Down Using Controller Switches	<b>51</b>
	Emergency Stop using External Switch	<b>51</b>
	Inverted Operation	<b>52</b>
	Special Use of Accessory Digital Inputs	<b>52</b>
	Using the Inputs to Activate the Buffered Output	<b>52</b>
	Using the Inputs to turn Off/On the Power MOSFET transistors	<b>52</b>
	Self-Test Mode	<b>53</b>
<b>SECTION 6</b>	Connecting Sensors and Actuators to Input/Outputs	<b>55</b>
	AX2550 Connections	<b>55</b>
	AX2550's Inputs and Outputs	<b>57</b>
	I/O List and Pin Assignment	<b>58</b>
	Connecting devices to Output C	<b>59</b>
	Connecting Switches or Devices to Input E	<b>60</b>
	Connecting Switches or Devices to Input F	<b>61</b>
	Connecting Switches or Devices to EStop/Invert Input	<b>62</b>
	Analog Inputs	<b>63</b>
	Connecting Position Potentiometers to Analog Inputs	<b>63</b>
	Connecting Tachometer to Analog Inputs	<b>64</b>
	Connecting External Thermistor to Analog Inputs	<b>66</b>
	Using the Analog Inputs to Monitor External Voltages	<b>67</b>
	Connecting User Devices to Analog Inputs	<b>67</b>
	Internal Voltage Monitoring Sensors	<b>68</b>
	Internal Heatsink Temperature Sensors	<b>68</b>
	Temperature Conversion C Source Code	<b>69</b>

<b>SECTION 7</b>	Installing, Connecting and Using the Encoder Module	<b>71</b>
	Optical Incremental Encoders Overview	<b>71</b>
	Recommended Encoder Types	<b>72</b>
	Installing the Encoder Module	<b>73</b>
	Connecting the Encoder	<b>75</b>
	Cable Length and Noise Considerations	<b>76</b>
	Motor - Encoder Polarity Matching	<b>76</b>
	Voltage Levels, Thresholds and Limit Switches	<b>76</b>
	Wiring Optional Limit Switches	<b>78</b>
	Wiring Limit Switches Without Encoders	<b>79</b>
	Effect of Limit Switches	<b>79</b>
	Using the Encoder Module to Measure Distance	<b>80</b>
	Using the Encoder to Measure Speed	<b>80</b>
	Using the Encoder to Track Position	<b>81</b>
	RS232 Communication with the Encoder Module	<b>82</b>
	Encoder Testing and Setting Using the PC Utility	<b>83</b>
<b>SECTION 8</b>	Closed Loop Position Mode	<b>85</b>
	Mode Description	<b>85</b>
	Selecting the Position Mode	<b>85</b>
	Position Sensor Selection	<b>86</b>
	Sensor Mounting	<b>86</b>
	Feedback Potentiometer wiring	<b>87</b>
	Feedback Potentiometer wiring in RC or RS232 Mode	<b>87</b>
	Feedback Potentiometer wiring in Analog Mode	<b>88</b>
	Analog Feedback on Single Channel Controllers	<b>89</b>
	Feedback Wiring in RC or RS232 Mode on Single Channel Controllers	<b>89</b>
	Feedback Wiring in Analog Mode on Single Channel Controllers	<b>89</b>
	Using Optical Encoders in Position Mode	<b>90</b>
	Sensor and Motor Polarity	<b>90</b>
	Encoder Error Detection and Protection	<b>91</b>
	Adding Safety Limit Switches	<b>91</b>
	Using Current Limiting as Protection	<b>93</b>
	Control Loop Description	<b>93</b>
	PID tuning in Position Mode	<b>94</b>
<b>SECTION 9</b>	Closed Loop Speed Mode	<b>97</b>
	Mode Description	<b>97</b>
	Selecting the Speed Mode	<b>97</b>
	Using Optical Encoder for Speed Feedback (AX2850 only)	<b>98</b>

	Tachometer or Encoder Mounting	98
	Tachometer wiring	98
	Speed Sensor and Motor Polarity	99
	Adjust Offset and Max Speed	100
	Control Loop Description	100
	PID tuning in Speed Mode	102
<b>SECTION 10</b>	Normal and Fault Condition LED Messages	103
	Use of the LED Display	103
	Motor Direction Status	104
	Fault Messages	105
	No Control	105
	Temporary Faults	106
	Permanent Faults	106
	Self-Test Display	106
<b>SECTION 11</b>	R/C Operation	109
	Mode Description	109
	Selecting the R/C Input Mode	110
	Connector I/O Pin Assignment (R/C Mode)	110
	R/C Input Circuit Description	111
	Supplied Cable Description	111
	Powering the Radio from the controller	112
	Connecting to a Separately Powered Radio	114
	Operating the Controller in R/C mode	114
	Reception Watchdog	115
	R/C Transmitter/Receiver Quality Considerations	116
	Joystick Deadband Programming	117
	Command Control Curves	118
	Left/Right Tuning Adjustment	118
	Joystick Calibration	118
	Automatic Joystick Calibration	119
	Activating the Accessory Outputs	120
	Data Logging in R/C Mode	121
<b>SECTION 12</b>	Analog Control and Operation	123
	Mode Description	123
	Connector I/O Pin Assignment (Analog Mode)	124
	Connecting to a Voltage Source	125
	Connecting a Potentiometer	125
	Selecting the Potentiometer Value	126



Analog Deadband Adjustment	127
Power-On Safety	128
Under Voltage Safety	128
Data Logging in Analog Mode	128

## **SECTION 13**

Serial (RS-232) Controls and Operation	131
Use and benefits of RS232	131
Connector I/O Pin Assignment (RS232 Mode)	132
Cable configuration	133
Extending the RS232 Cable	133
Communication Settings	134
Establishing Manual Communication with a PC	134
RS232 Communication with the Encoder Module	135
Entering RS232 from R/C or Analog mode	136
Data Logging String in R/C or Analog mode	136
RS232 Mode if default	137
Commands Acknowledge and Error Messages	137
Character Echo	137
Command Acknowledgement	137
Command Error	137
Watchdog time-out	137
RS-232 Watchdog	138
Controller Commands and Queries	138
Set Motor Command Value	139
Set Accessory Output	139
Query Power Applied to Motors	140
Query Amps from Battery to each Motor Channel	140
Query Analog Inputs	140
Query Heatsink Temperatures	141
Query Battery Voltages	141
Query Digital Inputs	142
Reset Controller	142
Accessing & Changing Configuration Parameter in Flash	143
Apply Parameter Changes	143
Flash Configuration Parameters List	144
Input Control Mode	145
Motor Control Mode	145
Amps Limit	146
Acceleration	147
Input Switches Function	147
RC Joystick or Analog Deadband	148
Exponentiation on Channel 1 and Channel 2	148
Left/Right Adjust	149
Default Encoder Time Base 1 and 2	149
Default Encoder Distance Divider	150

Default PID Gains	<b>150</b>
Joystick Min, Max and Center Values	<b>150</b>
Reading & Changing Operating Parameters at Runtime	<b>151</b>
Operating Modes Registers	<b>152</b>
Read/Change PID Values	<b>152</b>
PWM Frequency Register	<b>153</b>
Controller Status Register	<b>153</b>
Controller Identification Register	<b>154</b>
Current Amps Limit Registers	<b>154</b>
RS232 Encoder Command Set	<b>155</b>
Read Encoder Counter	<b>155</b>
Set/Reset Encoder Counters and Destination Registers	<b>155</b>
Read Speed	<b>156</b>
Read Distance	<b>157</b>
Read Speed/Distance	<b>157</b>
Read Encoder Limit Switch Status	<b>157</b>
Read / Modify Encoder Module Registers and Parameters	<b>158</b>
Register Description	<b>160</b>
Encoder Hardware ID code	<b>160</b>
Switch Status	<b>160</b>
Speed or Distance 1 or 2	<b>160</b>
Counter Read/Write Mailbox	<b>161</b>
Counter 1 and 2	<b>161</b>
Destination Register 1 and 2	<b>161</b>
Distance 1 and 2	<b>162</b>
Speed 1 and 2	<b>162</b>
Time Base 1 and 2	<b>162</b>
Encoder Threshold	<b>162</b>
Distance Divider	<b>162</b>
Counter Read Data Format	<b>163</b>
Encoder Testing and Setting Using the PC Utility	<b>163</b>
Automatic Switching from RS232 to RC Mode	<b>165</b>
Analog and R/C Modes Data Logging String Format	<b>166</b>
Data Logging Cables	<b>166</b>
Decimal to Hexadecimal Conversion Table	<b>167</b>
<b>SECTION 14</b>	
Configuring the Controller using the Switches	<b>171</b>
Programming Methods	<b>171</b>
Programming using built-in Switches and Display	<b>171</b>
Entering Programming Mode	<b>172</b>
Changing parameters	<b>173</b>
The Special Case of Joystick Calibration	<b>173</b>
Restoring factory defaults	<b>173</b>
Exiting the Parameter Setting Mode	<b>174</b>
Programmable Parameters List	<b>174</b>

<b>SECTION 15</b>	Using the Roborun Configuration Utility	<b>177</b>
	System Requirements	<b>177</b>
	Downloading and Installing the Utility	<b>177</b>
	Connecting the Controller to the PC	<b>178</b>
	Roborun Frame, Tab and Menu Descriptions	<b>179</b>
	Getting On-Screen Help	<b>180</b>
	Loading, Changing Controller Parameters	<b>181</b>
	Control Settings	<b>181</b>
	Power Settings	<b>182</b>
	Analog or R/C Specific Settings	<b>183</b>
	Closed Loop Parameters	<b>184</b>
	Encoder Setting and Testing	<b>184</b>
	Encoder Module Parameters Setting	<b>185</b>
	Exercising the Motors	<b>186</b>
	Viewing Encoder Data	<b>186</b>
	Running the Motors	<b>186</b>
	Logging Data to Disk	<b>189</b>
	Connecting a Joystick	<b>190</b>
	Using the Console	<b>191</b>
	Viewing and Logging Data in Analog and R/C Modes	<b>192</b>
	Loading and Saving Profiles to Disk	<b>192</b>
	Operating the AX2550 over a Wired or Wireless LAN	<b>193</b>
	Updating the Controller's Software	<b>194</b>
	Updating the Encoder Software	<b>195</b>
	Creating Customized Object Files	<b>195</b>
<b>SECTION 16</b>	Mechanical Specifications	<b>197</b>
	Mechanical Dimensions	<b>197</b>
	Mounting Considerations	<b>198</b>
	Thermal Considerations	<b>199</b>
	Attaching the Controller Directly to a Chassis	<b>199</b>
	Wire Dimensions	<b>199</b>
	Weight	<b>200</b>



# Important Safety Warnings

---

## Read this Section First

The AX2550 is a high power electronics device. Serious damage, including fire, may occur to the unit, motors, wiring and batteries as a result of its misuse. Please review the User's Manual for added precautions prior to applying full battery or full load power.

### **This product is intended for use with rechargeable batteries**

Unless special precautions are taken, damage to the controller and/or power supply may occur if operated with a power supply alone. See "Power Regeneration Considerations" on page 37 of the Users Manual. **Always keep the controller connected to the Battery.** Use the Power Control input to turn On/Off.

### **Do not Connect to a RC Radio with a Battery Attached**

Without proper protection, a battery attached to an RC Radio may inject its voltage directly inside the controller's sensitive electronics. See

### **Beware of Motor Runaway in Improperly Closed Loop**

Wiring or polarity errors between the feedback device and motor in position or closed loop position mode may cause the controller to runaway with no possibility to stop it until power is turned off.



# **AX2550**

## **Quick Start**

---

This section will give you the basic information needed to quickly install, setup and run your AX2550 controller in a minimal configuration. The AX2850 is a version of the AX2550 controller with the addition of Optical Encoder inputs.

---

### **What you will need**

For a minimal installation, gather the following components:

- One AX2550 Controller and its provided cables
- 12V to 40V high capacity, high current battery
- One or two brushed DC motors
- One R/C to DB15 connector (provided)
- Miscellaneous wires, connectors, fuses and switch

---

### **Locating the Switches, Wires and Connectors**

Take a moment to familiarize yourself with the controller's wires, switches and connectors.

The front side (shown in Figure 1) contains the buttons and display needed to operate and monitor the controller. The 15-pin connector provides the connection to the R/C radio, joystick or microcomputer, as well as connections to optional switches and sensors.

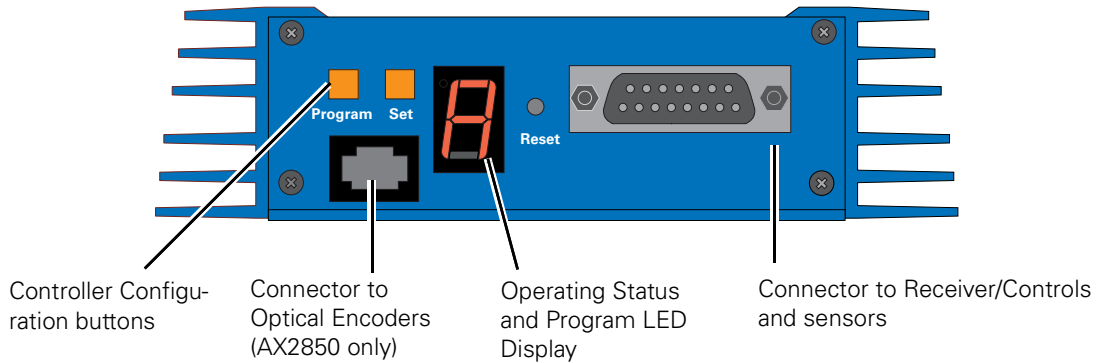


FIGURE 1. Front Controller Layout

At the back of the controller (shown in the figure below) are located all the wires that must be connected to the batteries and the motors.

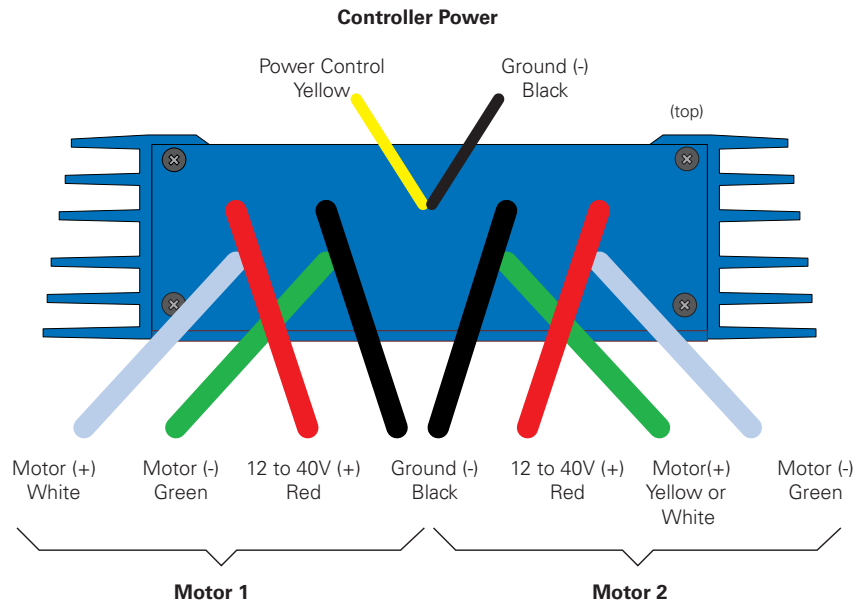
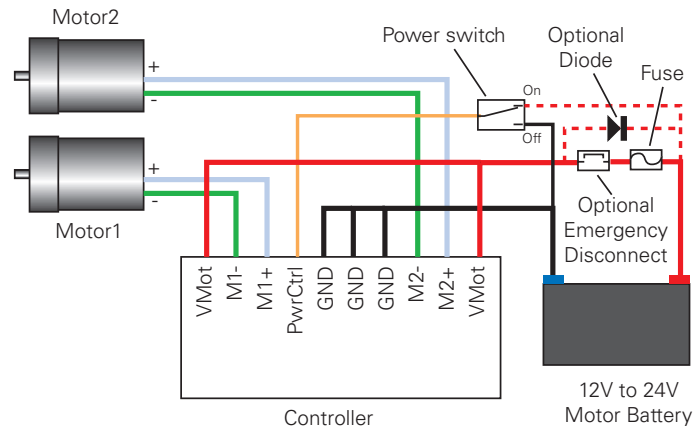


FIGURE 2. Rear Controller Layout



## Connecting to the Batteries and Motors

Connection to the batteries and motors is shown in the figure below and is done by connecting the set of wires coming out from the back of the controller.



Notes:

- The Battery Power connection are doubled in order to provide the maximum current to the controller. If only one motor is used, only one set of motor power cables needs to be connected.
- Typically, 1, 2 or 3 x 12V batteries are connected in series to reach 12, 24 or 36V respectively.
- The Power Control wire **MUST** be used to turn On and Off the controller.

FIGURE 3. AX2550 Electrical Power Wiring Diagram

1- Connect each motor to one of the two M+ and M- cables pairs. Make sure to respect the polarity, otherwise the motor(s) may spin in the opposite direction than expected

2- Connect the two thick black wires to the minus (-) terminal of the battery that will be used to power the motors. Connect the two thick red wires the two VMot terminals to the plus (+) terminal of the battery. The motor battery may be of 12 to 40 Volts. There is no need to insert a separate switch on Power cables, although one is suggested for Emergency disconnect. See "Controller Power" on page 30 for a detailed discussion and more wiring options.

Avoid extending the length of wires from the battery to the controller as the added inductance may cause damage to the controller when operating at high currents. Try extending the motor wires instead since the added inductance on the motor side of the controller is not harmful.

**The two red wires are connected to each other inside the controller. The same is true for the black wires. You should wire each pair together as shown in the diagram above.**

3- The yellow Power Control wire and the thin black wire **MUST** be connected to Ground to turn the Controller Off. For turning the controller On, even though the Power Control may be left floating, whenever possible pull it to an unfused 12V or higher voltage to keep the controller logic solidly On. You may use a separate battery to keep the controller alive as the main Motor battery discharges. Refer to the chapter "Connecting Power and Motors to

the Controller” on page 29 for more information about batteries and other connection options.

## Important Warning

**Do not rely on cutting power to the controller for it to turn off if the Power Control is left floating. If motors are spinning because the robot is pushed or because of inertia, they will act as generators and will turn the controller, possibly in an unsafe state. ALWAYS ground the Power Control wire to turn the controller Off and keep it Off.**

## Important Warning

**The controller includes large capacitors. When connecting the Motor Power Cables, a spark will be generated at the connection point. This is a normal occurrence and should be expected.**

## Connecting to the 15-pin Connector

The controller’s I/O are located on it’s standard 15-pin D-Sub Connector. The functions of some pins varies depending on controller model and operating mode. Pin assignment is found in the table below.

Pin	Signal		
	RC Mode	RS232 Mode	Analog Mode
1	2A Digital Output C (same as pin 9)		
2	TxData		
3	RC Ch1	RxDData	Unused
4	RC Ch 2	Digital Input F	
5	Ground Out		
6	Ground In (Unused on RevB Hardware)		
7	+5V In (Unused on RevB Hardware)		
8	Digital Input E (Not available when Encoder module is present) and Analog Input 4 (on RevB hardware only)		
9	2A Digital Output C (same as pin 1)		
10	Analog Input 2		
11	Analog Input 1		
12	Analog Input 3 (on RevB hardware)		
13	Ground Out		
14	+5V Out (100mA max.)		
15	Emergency Stop or Invert Switch input		

### Connecting the R/C Radio

Connect the R/C adapter cables to the controller on one side and to two or three channels on the R/C receiver on the other side. If present, the third channel is for activating the accessory outputs and is optional.

When operating the controller in “Separate” mode, the wire labelled Ch1 controls Motor1, and the wire labelled Ch2 controls Motor2.

When operating the controller in “Mixed” mode, Ch1 is used to set the robot’s speed and direction, while Ch2 is used for steering.

See “R/C Operation” on page 109 of the User’s Manual for a more complete discussion on R/C commands, calibration and other options.

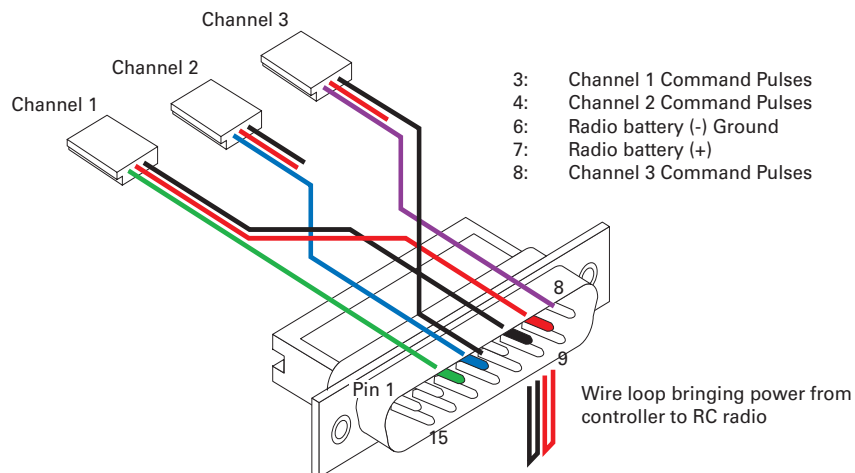


FIGURE 4. R/C connector wiring for 3 channels and battery elimination (BEC)

This wiring - with the wire loop uncut - assumes that the R/C radio will be powered by the AX2550 controller. Other wiring options are described in “R/C Operation” on page 109 of the User’s Manual.

### Important Warning

**Do not connect a battery to the radio when the wire loop is uncut. The RC battery voltage will flow directly into the controller and cause permanent damage if its voltage is higher than 5.5V.**

Connecting the optional channel 3 will enable you to turn on and off the accessory output. See “Connecting Sensors and Actuators to Input/Outputs” on page 55 and “Activating the Accessory Outputs” on page 120 of the User’s Manual.

## Powering On the Controller

**Important reminder:** There is no On-Off switch on the controller. You must insert a switch on the controller’s power wire as described in section “Connecting to the Batteries and Motors” on page 17.

To power the controller, center the joystick and trims on the R/C transmitter. In Analog mode, center the command potentiometer or joystick. Then turn on the switch that you have placed on the on the Power Control input.

If the R/C transmitter and/or receiver is powered off, the display on the controller will alternate the letters spelling “no ctrl” to indicate that it is On but is not receiving a control signal.



FIGURE 5. “no control” scroll message indicates no valid R/C signal is present

Turn the R/C transmitter On. The “no ctrl” scrolling message will disappear and the display will show steady patterns depending on the motor’s selected direction.

Move the joystick on the transmitter to activate the motors to the desired speed and direction.

See “R/C Operation” on page 109 of the User’s Manual for a detailed description of the many features and options available in the R/C mode.

## Button Operation

The AX2550 has three buttons: **Set**, **Program** and **Reset**. These buttons are not needed for normal operation, as the controller is immediately operational upon power up.

The **Reset** button will restart the controller. This button is recessed and you will need a paper clip to press it. Reset is also accomplished by turning the controller’s power Off and back On.

The **Set** and **Program** buttons have the following functions depending how and when they are pressed:

TABLE 1. AX2550 Buttons Function

Prog and Set button status	Function
Press and hold <b>Program</b> alone during reset or power up	Enter the Programming Mode.
Press and hold <b>Set</b> alone during reset of power up	Enter Self-Test mode. See “Self-Test Mode” on page 53 of the User’s Manual
Press and hold <b>Program</b> and <b>Set</b> together during reset or power up	Reset configuration parameters to factory default

TABLE 1. AX2550 Buttons Function

<b>Prog and Set button status</b>	<b>Function</b>
Press <b>Program</b> while Programming Mode	Accept previous parameter change and select next parameter
Press <b>Set</b> while in Programming mode	Change value of selected parameter
Press <b>Program</b> pressed alone during normal operation	No effect
Press <b>Set</b> alone during normal operation	No effect
Press <b>Program</b> and <b>Set</b> together during normal operation	Emergency stop

## Default Controller Configuration


Version 1.9b of the AX2550 software is configured with the factory defaults shown in the table below. Although Roboteq strives to keep the same parameters and values from one version to the next, changes may occur from one revision to the next. Make sure that you have the matching manual and software versions. These may be retrieved from the Roboteq web site. See "Configuring the Controller using the Switches" on page 171 of the User Manual for a complete configuration parameter list and their possible values.



TABLE 2. AX2550 Default Settings



<b>Parameter</b>	<b>Default Values</b>	<b>Letter</b>
Input Command mode:	(0) = R/C Radio mode	I
Motor Control mode	(0) = Separate A, B, speed control, open loop	C
Amp limit	(5) = 105A	A
Acceleration	(2) = medium-slow	S
Input switch function	(3) = no action	U
Joystick Deadband	(2) = 16%	d
Exponentiation on channel 1	(0) = Linear (no exponentiation)	E
Exponentiation on channel 2	(0) = Linear (no exponentiation)	F
Left / Right Adjust	(7) = no adjustment	L



Any one of the parameters listed in Table 2, and others not listed, can easily be changed either using the controller's buttons or the PC with the Roboteq Configuration Utility. See "Using the Roborun Configuration Utility" on page 177.



The example below shows how to use the buttons to select and change the Motor Control mode from "separate" to "mixed". See "Configuring the Controller using the Switches" on page 171 of the User's Manual for a complete list of all the AX2550's parameters and their meanings.



- Restart  
Press & hold **Prog** ▶ 

Press and hold the **Prog** button for 10 seconds while resetting or powering on the controller
- Program mode entered  
after 10 seconds ▶  

After 10 seconds, the controller will enter the programming mode and flash alternatively the current parameter (I= Input Mode) and its value (0= R/C mode).
- Press **Prog** to select  
next parameter ▶  

Press the **Prog** button to move to the next parameter (C= Motor Control Mode) and its value (0= Separate)
- Press **Set** to select  
next value for parameter ▶  

Press the **Set** button to change the parameter's value (1= Combined)
- Press **Prog** to store change  
and select next parameter ▶  

Press the **Prog** button record the change and move to the next parameter (A= Amps limit) and it's value (2= 75A)
- Reset controller  
to exit ▶  

Press the **Reset** button or power off/on the control to restart the controller using the new parameters.

## Connecting the controller to your PC using Roborun

Connecting the controller to your PC is not necessary for basic R/C operation. However, it is a very simple procedure that is useful for the following purposes:

- to Read and Set the programmable parameters with a user-friendly graphical interface
- to obtain the controller's software revision and date
- to send precise commands to the motors
- to read and plot real-time current consumption value
- Save captured parameters onto disk for later analysis
- to update the controller's software

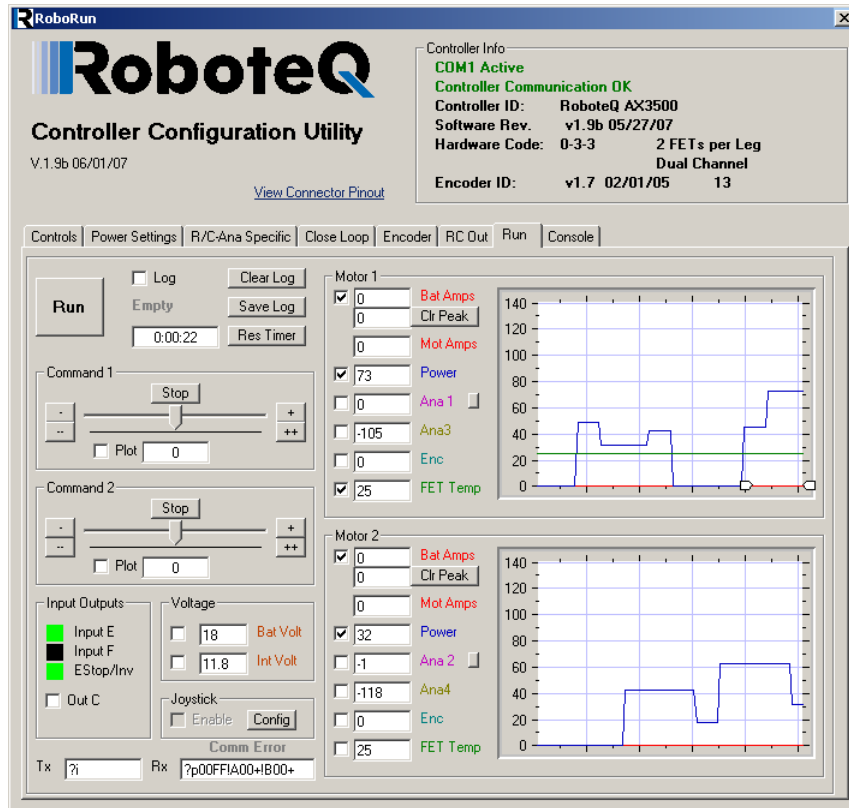


FIGURE 6. Roborun Utility screen layout

To connect the controller to your PC, use the provided cable. Connect the 15-pin connector to the controller. Connect the 9-pin connector to your PC's available port (typically COM1) - use a USB to serial adapter if needed. Apply power to the controller to turn it on.

Load your CD or download the latest revision of Roborun software from [www.Roboteq.com](http://www.Roboteq.com), install it on your PC and launch the program. The software will automatically establish communication with the controller, retrieve the software revision number and present a series of buttons and tabs to enable its various possibilities.

The intuitive Graphical User Interface will let you view and change any of the controller's parameters. The "Run" tab will present a number of buttons, dials and charts that are used for operating and monitoring the motors.

## Obtaining the Controller's Software Revision Number

One of the unique features of the AX2550 is the ability to easily update the controller's operating software with new revisions downloaded from Roboteq's web site at [www.roboteq.com](http://www.roboteq.com). This is useful for adding features and/or improving existing ones.

Each software version is identified with a unique number. Obtaining this number can be done using the PC connection discussed previously.

It is also possible to get the controller to display the software version number by following these simple steps

- Disconnect the power from the motor batteries. Power the controller only via the Power Control input.
- Press and hold the **Set** button while powering or resetting the controller.

The LED will display a sequence of two numerical digits and an optional letter separated by dashes as shown in the examples below.



FIGURE 7. Press and hold “Set” to display version number and enter self-test

After these digits are displayed, the controller will attempt to power the motors as part of the self test mode (see “Self-Test Mode” on page 53 of the User’s Manual for a more detailed explanation). This is why the **motor’s battery must be disconnected**. After about 30 seconds, the software revision number will be displayed again the cycle will repeat.

You will need to reset, or power down and up, the controller to exit and resume normal operation.

Now that you know your controller’s software version number, you will be able to see if a new version is available for download and installation from Roboteq’s web site and which features have been added or improved.

Installing new software is a simple and secure procedure, fully described in “Updating the Controller’s Software” on page 194 of the User’s Manual.

---

## Exploring further

By following this quick-start section, you should have managed to get your controller to operate in its basic modes within minutes of unpacking.

Each of the features mentioned thus far has numerous options which are discussed further in the complete User’s Manual, including:

- Self test mode
- Emergency stop condition
- Joystick calibration
- Using Inputs/Outputs
- Current limiting
- Closed Loop Operation
- Software updating
- and much more



---

# AX2550 Motor Controller Overview

---

Congratulations! By selecting Roboteq's AX2550 you have empowered yourself with the industry's most versatile, powerful and programmable DC Motor Controller for mobile robots. This manual will guide you step by step through its many possibilities.

---

## Product Description

The AX2550 is a highly configurable, microcomputer-based, dual-channel digital speed or position controller with built-in high power drivers. The controller is designed to interface directly to high power DC motors in computer controlled or remote controlled mobile robotics and automated vehicle applications.

The AX2550 controller can accept speed or position commands in a variety of ways: pulse-width based control from a standard Radio Control receiver, Analog Voltage commands, or RS-232 commands from a microcontroller or wireless modem.

The controller's two channels can be operated independently or can be combined to set the forward/reverse direction and steering of a vehicle by coordinating the motion on each side of the vehicle. In the speed control mode, the AX2550 can operate in open loop or closed loop. In closed loop operation, actual speed measurements from tachometers or optical encoders are used to verify that the motor is rotating at the desired speed and direction and to adjust the power to the motors accordingly.

The AX2550 can also be configured to operate as a precision, high torque servo controller. When connected to a potentiometer coupled to the motor assembly, the controller will command the motor to rotate up to a desired angular position. Depending on the DC motor's power and gear ratio, the AX2550 can be used to move or rotate steering columns or other physical objects with very high torque.

The AX2550 is fitted with many safety features ensuring a secure power-on start, automatic stop in case of command loss, over current protection on both channels, and overheat protection.

The motors are driven using high-efficiency Power MOSFET transistors controlled using Pulse Width Modulation (PWM) at 16kHz. The AX2550 power stages can operate from 12 to 40VDC and can sustain up to 120A of controlled current, delivering up to 4,800W (approximately 6 HP) of useful power to each motor.

The many programmable options of the AX2550 are easily configured using the supplied PC utility or one-touch Program and Set buttons and a 7-segment LED display. Once programmed, the configuration data are stored in the controller's non-volatile memory, eliminating the need for cumbersome and unreliable jumpers.

The AX2850 is the AX2550 controller fitted with a dual channel optical encoder input module. Optical Encoders allow precise motor speed and position measurement and enable advance robotic applications.

---

## Technical features

### **Fully Digital, Microcontroller-based Design**

- Multiple operating modes
- Fully programmable using either built-in switches and 7 segment display or through connection to a PC
- Non-volatile storage of user configurable settings
- Simple operation
- Software upgradable with new features

### **Multiple Command Modes**

- Radio-Control Pulse-Width input
- Serial port (RS-232) input
- 0-5V Analog Command input

### **Multiple Advanced Motor Control Modes**

- Independent operation on each channel
- Mixed control (sum and difference) for tank-like steering
- Open Loop or Closed Loop Speed mode
- Position control mode for building high power position servos
- Modes selectable independently for each channel

### **Automatic Joystick Command Corrections**

- Joystick min, max and center calibration
- Selectable deadband width
- Selectable exponentiation factors for each joystick
- 3rd R/C channel input for accessory output activation (disabled when encoder module present)

### **Special Function Inputs/Outputs**

- 2 Analog inputs. Used as:

- Tachometer inputs for closed loop speed control
- Potentiometer input for position (servo mode)
- Motor temperature sensor inputs
- External voltage sensors
- User defined purpose (RS232 mode only)
- 2 Extra analog inputs (on RevB hardware). Used as:
  - Potentiometer input for position while in analog command mode
  - User defined purpose (RS232 mode only)
- One Switch input configurable as
  - Emergency stop command
  - Reversing commands when running vehicle inverted
  - General purpose digital input
- One general purpose 24V, 2A output for accessories
- Up to 2 general purpose digital inputs

**Optical Encoder Inputs (AX2850 only)**

- Inputs for two Quadrature Optical Encoders
- up to 250kHz Encoder frequency per channel
- two 32-bit up-down counters
- Inputs may be shared with four optional limit switches per channel

**Internal Sensors**

- Voltage sensor for monitoring the main 12 to 40V battery system operation
- Voltage monitoring of internal 12V
- Temperature sensors on the heat sink of each power output stage
- Sensor information readable via RS232 port

**Low Power Consumption**

- On board DC/DC converter for single 12 to 40V battery system operation
- Optional backup power input for powering safely the controller if the motor batteries are discharged
- Max 200mA at 12V or 100mA at 24V idle current consumption
- Power Control wire for turning On or Off the controller from external microcomputer or switch
- No power consumed by output stage when motors are stopped
- Regulated 5V output for powering R/C radio. Eliminates the need for separate R/C battery

**High Efficiency Motor Power Outputs**

- Two independent power output stages
- Optional Single Channel operation at double the current
- Dual H bridge for full forward/reverse operation
- Ultra-efficient 2.5mOhm (1.25mOhm on HE version) ON resistance (RDSon) MOS-FET transistors
- Synchronous Rectification H Bridge

- 12 to 40 V operation
- High current 8 AWG cable sets for each power stages
- Temperature-based Automatic Current Limitation
  - 120A up to 15 seconds (per channel)
  - 100A up to 30 seconds
  - 80A extended
  - High current operation may be extended with forced cooling
- 250A peak Amps per channel
- 16kHz Pulse Width Modulation (PWM) output
- Auxiliary output for brake, clutch or armature excitation
- Heat sink extruded case

**Advanced Safety Features**

- Safe power on mode
- Optical isolation on R/C control inputs
- Automatic Power stage off in case of electrically or software induced program failure
- Overvoltage and Undervoltage protection
- Regeneration current limiting
- Watchdog for automatic motor shutdown in case of command loss (R/C and RS232 modes)
- Large, bright run/failure diagnostics on 7 segment LED display
- Programmable motor acceleration
- Built-in controller overheat sensor
- Emergency Stop input signal and button

**Data Logging Capabilities**

- 13 internal parameters, including battery voltage, captured R/C command, temperature and Amps accessible via RS232 port
- Data may be logged in a PC, PDA or microcomputer

**Sturdy and Compact Mechanical Design**

- Built from aluminum heat sink extrusion with mounting brackets
- Efficient heat sinking. Operates without a fan in most applications.
- 7" (178mm) long (excluding mounting brackets) by 5.5" wide (140mm) by 1.8" (40mm) high
- -20o to +85o C case operating environment
- 3.3 lbs (1500g)

## SECTION 4

# Connecting Power and Motors to the Controller

This section describes the AX2550 Controller's connections to power sources and motors.

## **Important Warning**

**Please follow the instructions in this section very carefully. Any problem due to wiring errors may have very serious consequences and will not be covered by the product's warranty.**

---

## **Power Connections**

The AX2550 has three Ground (black), two Vmot (red) power cables and a Power Control wire (yellow). The power cables are located at the back end of the controller. The various power cables are identified by their position, wire thickness and color: Red is positive (+), black is negative or ground (-).

The power connections to the batteries and motors are shown in the figure below.



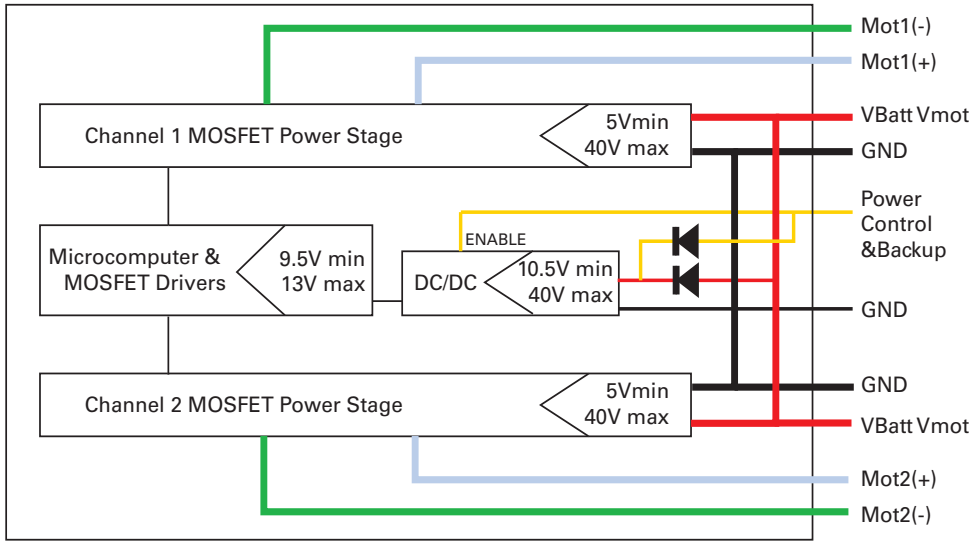


FIGURE 9. Representation of the AX2550's Internal Power Circuits

When powered only via the Power Control input, the controller will turn On but motors will not be able to turn until power is also present on the VMot wires

The Power Control input also serves as the Enable signal for the DC/DC converter. When floating or pulled to above 1V, the DC/DC converter is active and supplies the AX2550's microcomputer and drivers, thus turning it On. When the Power Control input is pulled to Ground, the DC/DC converter is stopped and the controller is turned Off.

The yellow Power Control wire and the thin black wire **MUST** be connected to Ground to turn the Controller Off. For turning the controller On, even though the Power Control may be left floating, whenever possible pull it to an unfused 12V or higher voltage to keep the controller logic solidly On. You may use a separate battery to keep the controller alive as the main Motor battery discharges.

The table below shows the state of the controller depending on the voltage applied to Power Control and Vmot.

TABLE 3. Controller Status depending on Power Control and VMot

Power Control input is connected to	And Main Battery Voltage is	Action
Ground	Any Voltage from 0V to 40V	Controller is Off
Floating	0V	Controller is Off. <b>Not Recommended Off Configuration.</b>
Floating	Between 8V and 10.5V	Controller Logic is On Power Stage is Disabled (under-voltage condition)
Floating	Between 10.5 and 40V	Controller is On. Power Stage is Active

TABLE 3. Controller Status depending on Power Control and VMot

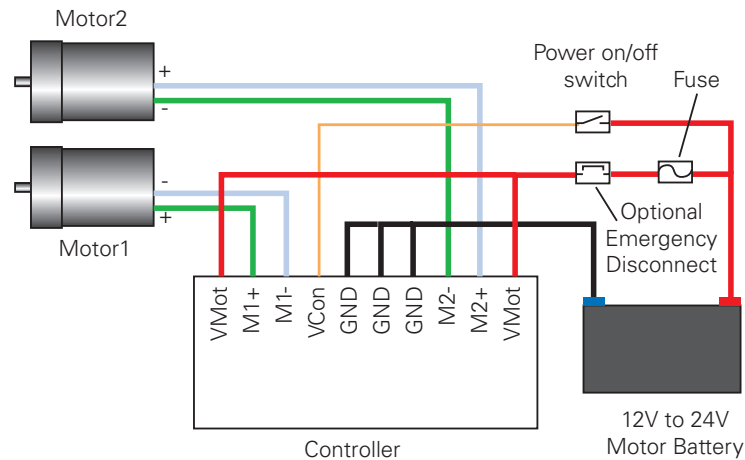
Power Control input is connected to	And Main Battery Voltage is	Action
10.5V to 40V	0V	Controller is On. Power Stage is Off
10.5V to 40V	1V to 40V	Controller is On. Power Stage is Active

All 3 ground (-) are connected to each other inside the controller. The two main battery wires are also connected to each other internally. However, you must never assume that connecting one wire of a given battery potential will eliminate the need to connect the other.

## Controller Powering Schemes

### Powering the Controller from a single Battery

The diagram on Figure 12 show how to wire the controller to a single battery circuit and how to turn power On and Off.



Notes:

- The Battery Power connection are doubled in order to provide the maximum current to the controller. If only one motor is used, only one set of motor power cables needs to be connected.
- Typically, 1, 2 or 3 x 12V batteries are connected in series to reach 12V, 24V or 36V respectively.

FIGURE 10. Powering the AX2550 from a single battery

Connect the two thick black wires to the minus (-) terminal of the battery that will be used to power the motors. Connect the two thick red wires to the plus (+) terminal of the battery. The motor battery may be of 12 to 40 Volts.



There is no need to insert a separate switch on Power cables, although for safety reasons, it is highly recommended that a way of quickly disconnecting the Motor Power be provided in the case of loss of control and all of the AX2550 safety features fail to activate.

**The two red wires are connected to each other inside the controller. The same is true for the black wires. You should wire each pair together as shown in the diagram above.**

The yellow Power Control wire and the thin black wire **MUST** be connected to Ground to turn the Controller Off. When the controller is Off, the output transistors are in the Off position and no power is drawn on VMot.

For turning the controller On, even though the Power Control may be left floating, whenever possible pull it to an unfused 12V or higher voltage to keep the controller logic solidly On. In applications where the motors could be made to run through external force (electric vehicle going downhill, for example), and generate 40V or more, a diode should be placed across the fuse & emergency switch to provide a path, under all circumstances, for the regeneration current. See "Power Regeneration Considerations" on page 37.

## **Important Warning**

**Do not rely on cutting power to the controller for it to turn off if the Power Control is left floating. If motors are spinning because the robot is pushed or because of inertia, they will act as generators and will turn the controller On, possibly in an unsafe state. ALWAYS ground the Power Control wire to turn the controller Off and keep it Off.**

## **Important Warning**

**On versions of the AX2550 with PCB revision number lower than 5.2, the backup power supply applied on the Power Control wire must NEVER EXCEED 13V. Permanent damage may otherwise occur. The PCB revision number can be found on the sticker on the case's bottom.**

## **Important Warning**

**Always connect the thick black wires to ground BEFORE connecting the red wires to the battery. If the motors are activated while the connection to ground is done solely via the thin black wire (or the ground wire in the RS232 cable), the ground wire will overheat and/or melt .**

## **Powering the Controller Using a Main and Backup Battery**

In typical applications, the main motor batteries will get eventually weaker and the voltage will drop below the level needed for the internal microcomputer to properly operate. For all professional applications it is therefore recommended to add a separate 12V (to 40V) power supply to ensure proper powering of the controller under any conditions. This dual battery configuration is highly recommended in 12V systems.

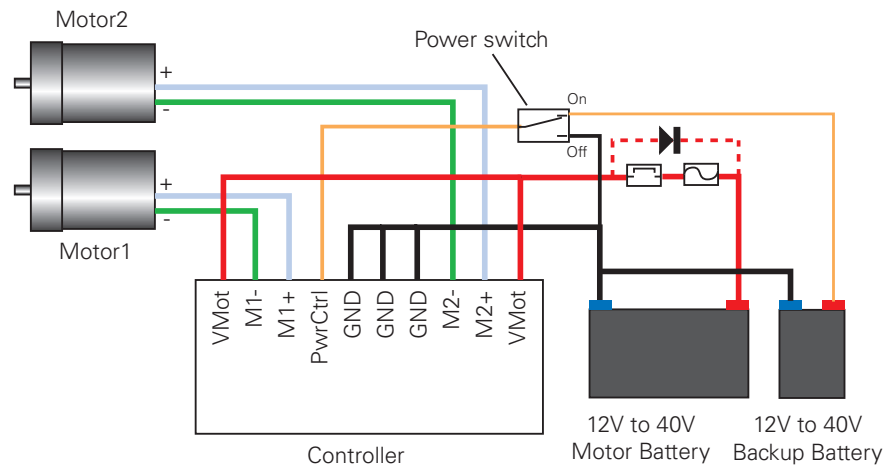


FIGURE 11. Powering the AX2550 with a Main and Backup Supply

## Important Warning

Unless you can ensure a steady 12V to 40V voltage in all conditions, it is recommended that the battery used to power the controller's electronics be separate from the one used to power the motors. This is because it is very likely that the motor batteries will be subject to very large current loads which may cause the voltage to eventually dip below 12V as the batteries' charge drops. The separate backup power supply should be connected to the Power Control input.

## Connecting the Motors

Connecting the motors is simply done by connecting each motor terminal to the M1+ (M2+) and M1- (M2-) wires. Which motor terminal goes to which of the + or - controller output is typically determined empirically.

After connecting the motors, apply a minimal amount of power using the Roborun PC utility with the controller configured in Open Loop speed mode. Verify that the motor spins in the desired direction. Immediately stop and swap the motor wires if not.

In Closed Loop Speed or Position mode, beware that the motor polarity must match this of the feedback. If it does not, the motors will runaway with no possibility to stop other than switching Off the power. The polarity of the Motor or off the feedback device may need to be changed.

## Important Warning

Make sure that your motors have their wires isolated from the motor casing. Some motors, particularly automotive parts, use only one wire, with the other connected to the motor's frame.

**If you are using this type of motor, make sure that it is mounted on isolators and that its casing will not cause a short circuit with other motors and circuits which may also be inadvertently connected to the same metal chassis.**

## Single Channel Operation

The AX2550's two channel outputs can be paralleled as shown in the figure below so that they can drive a single load with twice the power. To perform in this manner, the controller's Power Transistor that are switching in each channel must be perfectly synchronized. Without this synchronization, the current will flow from one channel to the other and cause the destruction of the controller.

The controller may be ordered with the -SC (Single Channel) suffix. This version incorporates a hardware setting inside the controller which ensures that both channels switch in a synchronized manner and respond to commands sent to channel 1.

**Warning:**

Use this wiring only with -SC versions (Single Channel) of the controller

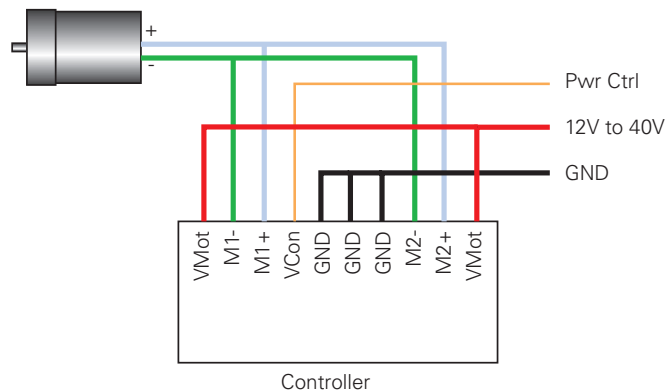


FIGURE 12. Wiring for Single Channel Operation

## Converting the AX2550 to Single Channel

The AX2550 can be easily modified into a Single Channel version by placing a jumper on the PCB. This step must be undertaken only if you have the proper tooling and technical skills.

- Disconnect the controller from power
- Open the controller's case by removing the front bracket and sliding the cover off
- Depending on the board's revision, place a drop of solder on the PCB jumper pad shown in Figure 13 or insert a jumper as shown in Figure 14.
- Place the cover and bracket back

Before paralleling the outputs,

- Place the load on channel 1 and verify that it is activated by commands on channel 1.

- Then place the load on channel 2 and verify that is also activated by commands on channel 1.
- Commands on channel 2 should have no effects on either output.

It will be safe to wire in parallel the controller's outputs only after you have verified that both outputs react identically to channel 1 commands.

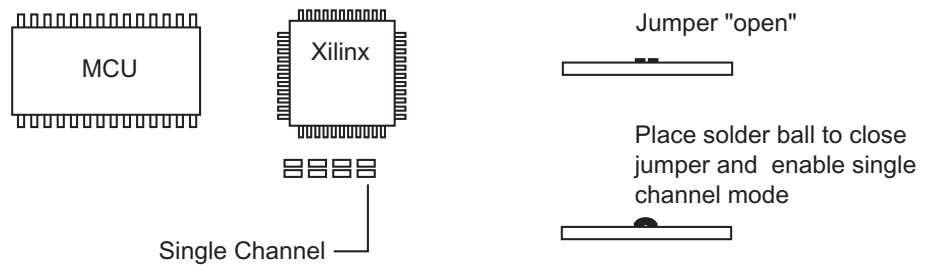


FIGURE 13. Solder Jumper setting for Single Channel Operation

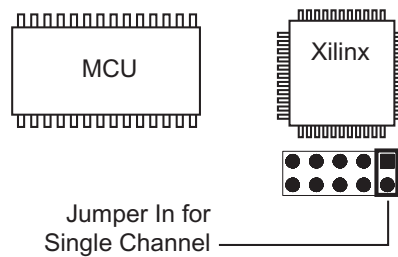


FIGURE 14. Jumper setting for Single Channel Operation - RevB Hardware

## Power Fuses

For low Amperage applications (below 30A per motor), it is recommended that a fuse be inserted in series with the main battery circuit as shown in the Figure 10 on page 32.

The fuse will be shared by the two output stages and therefore must be placed before the Y connection to the two power wires. Fuse rating should be the sum of the expected current on both channels. Note that automotive fuses are generally slow will be of limited effectiveness in protecting the controller and may be omitted in high current application. The fuse will mostly protect the wiring and battery against after the controller has failed.

## Important Warning

**Fuses are typically slow to blow and will thus allow temporary excess current to flow through them for a time (the higher the excess current, the faster the fuse will blow). This characteristic is desirable in most cases, as it will allow motors to draw surges**

**during acceleration and braking. However, it also means that the fuse may not be able to protect the controller.**

---

## **Wire Length Limits**

The AX2550 regulates the output power by switching the power to the motors On and Off at high frequencies. At such frequencies, the wires' inductance produces undesirable effects such as parasitic RF emissions, ringing and overvoltage peaks. The controller has built-in capacitors and voltage limiters that will reduce these effects. However, should the wire inductance be increased, for example by extending the wire length, these effects will be amplified beyond the controller's capability to correct them. This is particularly the case for the main battery power wires (thick red and black cables).

## **Important Warning**

**Avoid extending the black and red Motor Power wires beyond their original length, as the added inductance may cause damage to the controller when operating at high currents. Try extending the motor wires instead since the added inductance is less harmful on this side of the controller.**

If the controller must be located at a longer distance, the effects of the wire inductance may be reduced by using one or more of the following techniques:

- Twisting the power and ground wires over the full length of the wires
- Use the vehicle's metallic chassis for ground and run the positive wire along the surface
- Add a capacitor (5,000uF or higher) near the controller

---

## **Electrical Noise Reduction Techniques**

As discussed in the above section, the AX2550 uses fast switching technology to control the amount of power applied to the motors. While the controller incorporates several circuits to keep electrical noise to a minimum, additional techniques can be used to keep the noise low when installing the AX2550 in an application. Below is a list of techniques you can try to keep noise emission low:

- Keep wires as short as possible
- Loop wires through ferrite cores
- Add snubber R/C circuit at motor terminals
- Keep controller, wires and battery enclosed in metallic body

---

## **Power Regeneration Considerations**

When a motor is spinning faster than it would normally at the applied voltage, such as when moving downhill or decelerating, the motor acts like a generator. In such cases, the current will flow in the opposite direction, back to the power source.

It is therefore essential that the AX2550 be connected to rechargeable batteries. If a power supply is used instead, the current will attempt to flow back in the power supply during regeneration, potentially damaging it and/or the controller.

Regeneration can also cause potential problems if the battery is disconnected while the motors are still spinning. In such a case, and depending on the command level applied at that time, the regenerated current will attempt to flow back to the battery. Since none is present, the voltage will rise to potentially unsafe levels. The AX2550 includes an overvoltage protection circuit to prevent damage to the output transistors (see "Overvoltage Protection" on page 38). However, if there is a possibility that the motor could be made to spin and generate a voltage higher than 40V, a path to the battery must be provided, even after a fuse is blown. This can be accomplished by inserting a diode across the fuse as shown in Figure 10 on page 32.

Please download the Application Note "Understanding Regeneration" from the [www.roboteq.com](http://www.roboteq.com) for an in-depth discussion of this complex but important topic.

## **Important Warning**

**Use the AX2550 only with a rechargeable battery as supply to the Motor Power wires (thick black and red wires). If a transformer or power supply is used, damage to the controller and/or power supply may occur during regeneration. See "Using the Controller with a Power Supply" on page 39 for details.**

## **Important Warning**

**Avoid switching Off or cutting open the main power cables (thick black and red wires) while the motors are spinning. Damage to the controller may occur.**

---

## **Overvoltage Protection**

The AX2550 includes a battery voltage monitoring circuit that will cause the output transistors to be turned Off if the main battery voltage applied on the thick red and black wires rises above 43V.

This protection is designed to prevent the voltage created by the motors during regeneration to be "amplified" to unsafe levels by the switching circuit.

The controller will resume normal operation when the measured voltage drops below 43V.

---

## **Undervoltage Protection**

In order to ensure that the power MOSFET transistors are switched properly, the AX2550 monitors the internal 12V power supply that is used by the MOSFET drivers. If the internal voltage drops below 10V, the controller's output stage is turned Off. The rest of the controller's electronics, including the microcomputer, will remain operational as long as the internal voltage is above 8V.

The internal voltage will be the output of the DC/DC converter which will be a solid 12V as long as either of the main battery or backup voltage is higher than 12.5V. If the main and

backup voltage drop below 12.V, the DC/DC converter's output will be approximately 0.5V lower than the highest input.

---

## Using the Controller with a Power Supply

Using a transformer or a switching power supply is possible but requires special care, as the current will want to flow back from the motors to the power supply during regeneration. As discussed in "Power Regeneration Considerations" on page 37, if the supply is not able to absorb and dissipate regenerated current, the voltage will increase until the over-voltage protection circuit cuts off the motors. While this process should not be harmful to the controller, it may be to the power supply, unless one or more of the protective steps below is taken:

- Use a power supply that will not suffer damage in case a voltage is applied at its output that is higher than the transformer's own output voltage. This information is seldom published in commercial power supplies, so it is not always possible to obtain positive reassurance that the supply will survive such a condition.
- Avoid deceleration that is quicker than the natural deceleration due to the friction in the motor assembly (motor, gears, load). Any deceleration that would be quicker than natural friction means that braking energy will need to be taken out of the system, causing a reverse current flow and voltage rise. See "Programmable Acceleration" on page 47.
- Place a battery in parallel with the power supply output. This will provide a reservoir into which regeneration current can flow. It will also be very helpful for delivering high current surges during motor acceleration, making it possible to use a lower current power supply. Batteries mounted in this way should be connected for the first time only while fully charged and should not be allowed to discharge. The power supply will be required to output unsafe amounts of current if connected directly to a discharged battery. Consider using a decoupling diode on the power supply's output to prevent battery or regeneration current to flow back into the power supply.
- Place a resistive load in parallel with the power supply, with a circuit to enable that load during regeneration. This solution is more complex but will provide a safe path for the braking energy into a load designed to dissipate it. To prevent current from flowing from the power supply into the load during normal operation, an active switch would enable the load when the voltage rises above the nominal output of the power supply.





## SECTION 5

# General Operation

This section discusses the controller's normal operation in all its supported operating modes.

---

## Basic Operation

The AX2550's operation can be summarized as follows:

- Receive commands from a radio receiver, joystick or a microcomputer
- Activate the motors according to the received command
- Perform continuous check of fault conditions and adjust actions accordingly

Multiple options are available for each of the above listed functions which can be combined to produce practically any desired mobile robot configuration.

---

## Input Command Modes

The controller will accept commands from one of the following sources

- R/C radio
- Serial data (RS232)
- Analog signal (0 to 5V)

A detailed discussion on each of these modes and the available commands is provided in the following dedicated chapters: "R/C Operation" on page 109, "Serial (RS-232) Controls and Operation" on page 131, and "Analog Control and Operation" on page 123.

The controller's factory default mode is R/C radio. The mode can be changed using any of the methods described in "Programming using built-in Switches and Display" on page 171 and "Loading, Changing Controller Parameters" on page 181.

## Selecting the Motor Control Modes

For each motor, the AX2550 supports multiple motion control modes. The controller’s factory default mode is Open Loop Speed control for each motor. The mode can be changed using any of the methods described in “Programming using built-in Switches and Display” on page 171 and “Loading, Changing Controller Parameters” on page 181.

### Open Loop, Separate Speed Control

In this mode, the controller delivers an amount of power proportional to the command information. The actual motor speed is not measured. Therefore the motors will slow down if there is a change in load as when encountering an obstacle and change in slope. This mode is adequate for most applications where the operator maintains a visual contact with the robot.

In the separate speed control mode, channel 1 commands affect only motor 1, while channel 2 commands affect only motor 2. This is illustrated in Figure 15 below.

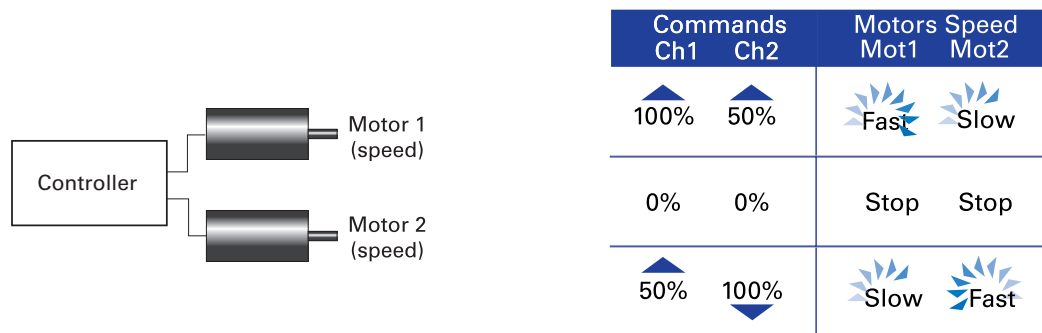


FIGURE 15. Examples of effect of commands to motors in separate mode

### Open Loop, Mixed Speed Control

This mode has the same open loop characteristics as the previously described mode. However, the two commands are now mixed to create tank-like steering when one motor is used on each side of the robot: Channel 1 is used for moving the robot in the forward or reverse direction. Channel 2 is used for steering and will change the balance of power on each side to cause the robot to turn.

Figure 16 below illustrates how the mixed mode works.

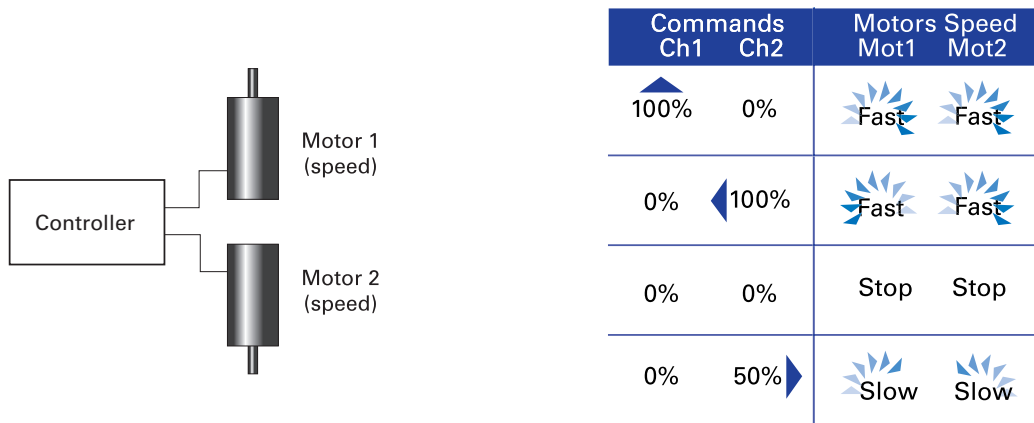


FIGURE 16. Effect of commands to motors examples in mixed mode

### Closed Loop Speed Control

In this mode, illustrated in Figure 18, an analog tachometer or an optical encoder (AX2850 only) is used to measure the actual motor speed. If the speed changes because of changes in load, the controller automatically compensates the power output. This mode is preferred in precision motor control and autonomous robotic applications. Details on how to wire the tachometer can be found in "Connecting Tachometer to Analog Inputs" on page 64. Closed Loop Speed control operation is described in "Closed Loop Speed Mode" on page 97.

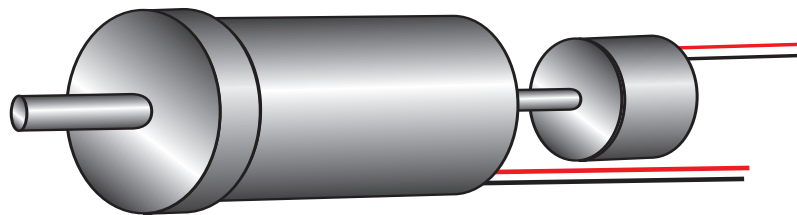


FIGURE 17. Motor with tachometer or Encoder for Closed Loop Speed operation

### Close Loop Position Control

In this mode, illustrated in Figure 18, the axle of a geared down motor is coupled to a potentiometer that is used to compare the angular position of the axle versus a desired position. This AX2550 feature makes it possible to build ultra-high torque "jumbo servos" that can be used to drive steering columns, robotic arms, life-size models and other heavy loads. Details on how to wire the position sensing potentiometers and operating in this mode can be found in "Closed Loop Position Mode" on page 85.

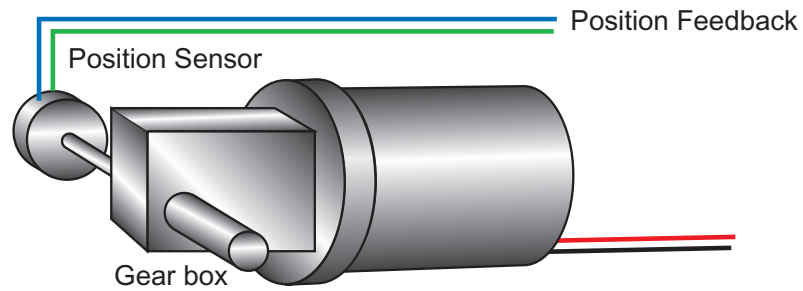


FIGURE 18. Motor with potentiometer assembly for Position operation

## User Selected Current Limit Settings

The AX2550 has current sensors at each of its two output stages. Every 16 ms, this current is measured and a correction to the output power level is applied if higher than the user preset value.

The current limit may be set using the controller's switches or the supplied PC utility. Using the switches, 7 limits may be selected as shown in the table below.

TABLE 4. Current limit settings using the switches

Setting	Continuous High Amps
0	30A
1	45A
2	60A
3	75A
4	90A
5 - default	105A
6	120A

Using the PC utility it is possible to set the limit with a 1A granularity from 15A to 120A

During normal operation, current limiting is further enhanced by the techniques described in the following sections.

## Temperature-Based Current Limitation

The AX2550 features active current limitation that uses a combination of a user defined preset value (discussed above) which in turn may be reduced automatically based on measured operating temperature. This capability ensures that the controller will be able to work safely with practically all motor types and will adjust itself automatically for the various load conditions.

When the measured temperature reaches 80oC, the controller’s maximum current limit begins to drop to reach 0A at 100oC. Above 100oC, the controller’s power stage turns itself off completely.

TABLE 5. Effect of Case temperature on Max Amps Limit

<b>Temperature</b>	<b>Max Amps</b>
Below 80 oC	120A
80 oC	120A
85 oC	80A
90 oC	40A
95 oC	20A
100 oC	0
Above 100 oC	Both Power Stages OFF

The numbers in the table are the max Amps allowed by the controller at a given temperature point. If the Amps limit is manually set to a lower value, then the controller will limit the current to the lowest of the manual and temperature-adjusted max values.

This capability ensures that the controller will be able to work safely with practically all motor types and will adjust itself automatically for the various load and environmental conditions. The time it takes for the heat sink’s temperature to rise depends on the current output, ambient temperature, and available air flow (natural or forced).

Note that the measured temperature is measured on the heat sink near the Power Transistors and will rise and fall faster than the outside surface.

---

## **Battery Current vs. Motor Current**

The controller measures and limits the current that flows from the battery. Current that flows through the motor is typically higher. This counter-intuitive phenomenon is due to the “flyback” current in the motor’s inductance. In some cases, the motor current can be extremely high, causing heat and potentially damage while battery current appears low or reasonable.

The motor’s power is controlled by varying the On/Off duty cycle of the battery voltage 16,000 times per second to the motor from 0% (motor off) to 100 (motor on). Because of the flyback effect, during the Off time current continues to flow at nearly the same peak - and not the average - level as during the On time. At low PWM ratios, the peak current - and therefore motor current - can be very high as shown in Figure 20, “Instant and average current waveforms,” on page 46.

The relation between Battery Current and Motor current is given in the formula below:

$$\text{Motor Current} = \text{Battery Current} / \text{PWM ratio}$$

Example: If the controller reports 10A of battery current while at 10% PWM, the current in the motor is  $10 / 0.1 = 100A$ .

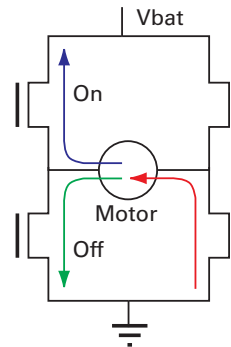


FIGURE 19. Current flow during operation

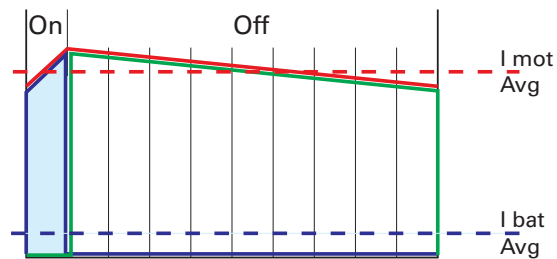


FIGURE 20. Instant and average current waveforms

The relation between Battery Current and Motor current is given in the formula below:

$$\text{Motor Current} = \text{Battery Current} / \text{PWM Ratio}$$

Example: If the controller reports 10A of battery current while at 10% PWM, the current in the motor is  $10 / 0.1 = 100\text{A}$ .

## Important Warning

**Do not connect a motor that is rated at a higher current than the controller. While the battery current will never exceed the preset Amps limit, that limit may be reached at a PWM cycle lower than 100% resulting in a higher and potentially unsafe level through the motor and the controller.**

## Regeneration Current Limiting

The AX2550's current sensor is capable of measuring current in the reverse flow (regeneration). Using this capability, the controller will automatically relax the braking effect of the power output stage to keep the regeneration current within safe values. Because of the controller's high current handling capabilities, this protection mechanism activates only when abrupt deceleration are applied to high-inertia, ultra-low impedance motors.

## Programmable Acceleration

When changing speed command, the AX2550 will go from the present speed to the desired one at a user selectable acceleration. This feature is necessary in order to minimize the surge current and mechanical stress during abrupt speed changes.

This parameter can be changed by using the controller's front switches or using serial commands. When configuring the controller using the switches (see "Configuring the Controller using the Switches" on page 171), acceleration can be one of 6 available preset values, from very soft(0) to very quick (6). The AX2550's factory default value is medium soft (2).

When using the serial port, acceleration can be one of 24 possible values, selectable using the Roborun utility or entering directly a value in the MCU's configuration EEPROM. Table 6 shows the corresponding acceleration for all Switch and RS232 settings.

Numerically speaking, each acceleration value corresponds to a fixed percentage speed increment, applied every 16 milliseconds. The value for each setting is shown in the table below.

TABLE 6. Acceleration setting table

<b>Acceleration Setting Using RS232</b>	<b>Acceleration Setting Using Switches</b>	<b>%Acceleration per 16ms</b>	<b>Time from 0 to max speed</b>
30 Hex		0.78%	2.05 seconds
20 Hex		1.56%	1.02 seconds
10 Hex		2.34%	0.68 second
00 Hex	0	3.13%	0.51 second
31 Hex		3.91%	0.41 second
21 Hex		4.69%	0.34 second
11 Hex		5.47%	0.29 second
01 Hex	1	6.25%	0.26 second
32 Hex	-	7.03%	0.23 second
22 Hex	-	7.81%	0.20 second
12 Hex	-	8.59%	0.19 second
02 Hex	2 (default)	9.38%	0.17 second
33 Hex	-	10.16%	0.16 second
23 Hex	-	10.94%	0.15 second
13 Hex	-	11.72%	0.14 second
03 Hex	3	12.50%	0.128 second
34 Hex	-	13.28%	0.120 second
24 Hex	-	14.06%	0.113 second
14 Hex	-	14.84%	0.107 second
04 Hex	4	15.63%	0.102 second
35 Hex	-	16.41%	0.097 second
25 Hex	-	17.19%	0.093 second

TABLE 6. Acceleration setting table

Acceleration Setting Using RS232	Acceleration Setting Using Switches	%Acceleration per 16ms	Time from 0 to max speed
15 Hex	-	17.97%	0.089 second
05 Hex	5	18.75%	0.085 second

When configuring the acceleration parameter using the Roborun utility, four additional acceleration steps can be selected between the six ones selectable using the switch, extending the slowest acceleration to 2.04 seconds from 0 to max speed. See "Power Settings" on page 182 for details on how to configure this parameter using Roborun.

## Important Warning

**Depending on the load's weight and inertia, a quick acceleration can cause considerable current surges from the batteries into the motor. A quick deceleration will cause an equally large, or possibly larger, regeneration current surge. Always experiment with the lowest acceleration value first and settle for the slowest acceptable value.**

## Command Control Curves

The AX2550 can also be set to translate the joystick or RS232 motor commands so that the motors respond differently whether or not the joystick is near the center or near the extremes.

The controller can be configured to use one of 5 different curves independently set for each channel.

The factory default curve is a "linear" straight line, meaning that after the joystick has moved passed the deadband point, the motor's speed will change proportionally to the joystick position.

Two "exponential" curves, a weak and a strong, are supported. Using these curves, and after the joystick has moved past the deadband, the motor speed will first increase slowly, increasing faster as the joystick moves near the extreme position. Exponential curves allow better control at slow speed while maintaining the robot's ability to run at maximum speed.

Two "logarithmic" curves, a weak and a strong, are supported. Using these curves, and after the joystick has moved past the deadpoint, the motor speed will increase rapidly, and then increase less rapidly as the joystick moves near the extreme position.

The graph below shows the details of these curves and their effect on the output power as the joystick is moved from its center position to either extreme. The graph is for one joystick only. The graph also shows the effect of the deadband setting.



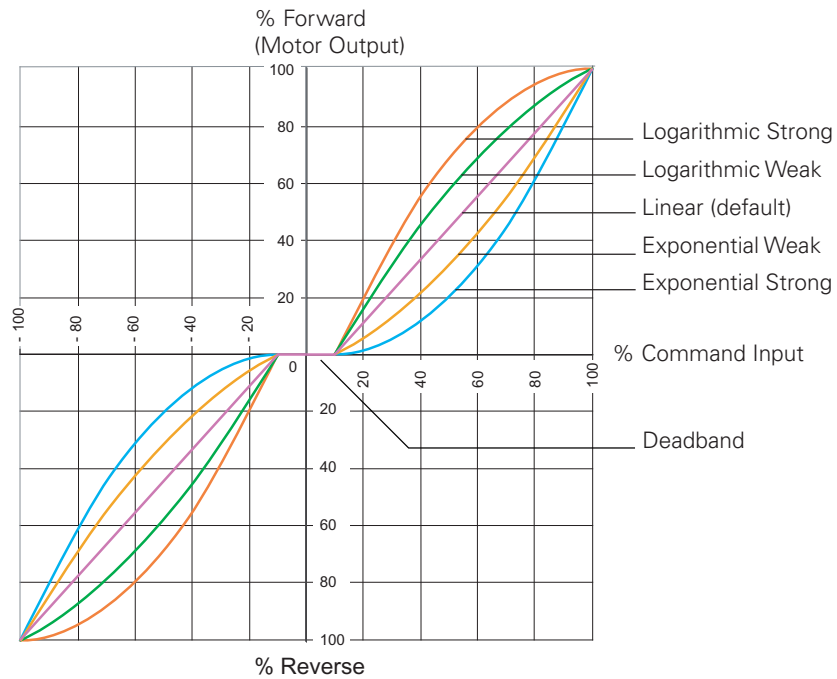


FIGURE 21. Exponentiation curves

The AX2550 is delivered with the “linear” curves selected for both joystick channels. To select different curves, the user will need to change the values of “E” (channel 1) and “F” (channel 2) according to the table below. Refer to the chapter “Configuring the Controller using the Switches” on page 171 or “Using the Roborun Configuration Utility” on page 177 for instructions on how to program parameters into the controller.

TABLE 7. Exponent selection table

Exponentiation Parameter Value	Selected Curve
E or F = 0	Linear (no exponentiation) - default value
E or F = 1	strong exponential
E or F = 2	normal exponential
E or F = 3	normal logarithmic
E or F = 4	strong logarithmic

## Left / Right Tuning Adjustment

By design, DC motors will run more efficiently in one direction than the other. In most situations this is not noticeable. In others, however, it can be an inconvenience. When operating in open loop speed control, the AX2550 can be configured to correct the speed in one direction versus the other by as much as 10%. Unlike the Joystick center trimming tab that

is found on all R/C transmitters, and which is actually an offset correction, the Left/Right Adjustment is a true multiplication factor as shown in Figure 22

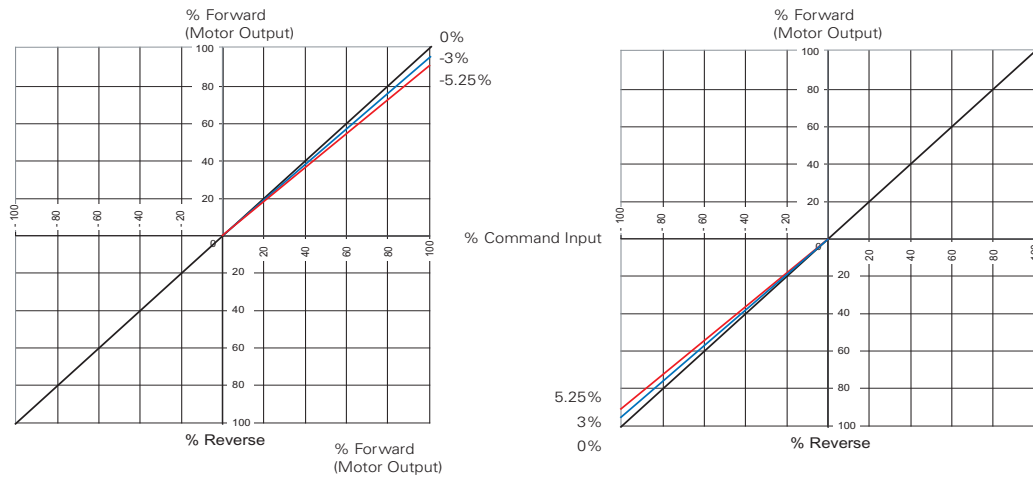


FIGURE 22. Left Right adjustment curves

The curves on the left show how a given forward direction command value will cause the motor to spin 3 or 5.25% slower than the same command value applied in the reverse direction. The curves on the right show how the same command applied to the forward direction will cause the motor to spin 3 to 5.25% faster than the same command applied in the reverse direction. Note that since the motors cannot be made to spin faster than 100%, the reverse direction is the one that is actually slowed down.

In applications where two motors are used in a mixed mode for steering, the Left/Right Adjustment parameter may be used to make the robot go straight in case of a natural tendency to steer slightly to the left or to the right.

The Left/Right adjustment parameter can be set from -5.25% to +5.25% in seven steps of 0.75%. See "Programmable Parameters List" on page 174 and "Loading, Changing Controller Parameters" on page 181 for details on how to adjust this parameter.

The Left/Right adjustment is performed in addition to the other command curves described in this section. This adjustment is disabled when the controller operates in any of the supported closed loop modes.

TABLE 8. Left/Right Adjustment Parameter selection

Parameter Value	Speed Adjustment	Parameter Value	Speed Adjustment
		7	None (default)
0	-5.25%	8	0.75%
1	-4.5%	9	1.5%
2	-3.75%	10	2.25%
3	-3%	11	3%
4	-2.25%	12	3.75%

TABLE 8. Left/Right Adjustment Parameter selection

Parameter Value	Speed Adjustment	Parameter Value	Speed Adjustment
5	-1.5%	12	4.5%
6	-0.75%	14	5.25%

## Activating Brake Release or Separate Motor Excitation

The controller may be configured so that the Output C will turn On whenever one of the two motors is running. This feature is typically used to activate the mechanical brake release sometimes found on motors for personal mobility systems. Likewise, this output can be used to turn on or off the winding that creates the armature's magnetic field in a separate excitation motor. This function is disabled by default and may be configured using the Roborun PC utility. See "Loading, Changing Controller Parameters" on page 181. See "Connecting devices to Output C" on page 59 for details on how to connect to the output.

## Emergency Shut Down Using Controller Switches

In case of emergency, it is possible to cause the controller to cut off the power to the motors by depressing the Program and Set buttons simultaneously as shown in the figure below. The controller will stop immediately without delay.

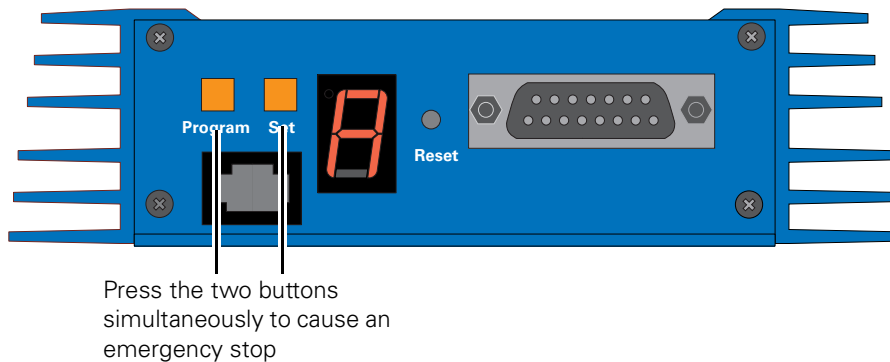


FIGURE 23. Activating an Emergency Stop

**After an Emergency Stop condition, the controller must be reset or powered Off and On to resume normal operation.**

## Emergency Stop using External Switch

An external switch can be added to the AX2550 to allow the operator to stop the controller's output in case of emergency. This controller input can be configured as the "Inverted" detection instead of Emergency Stop. The factory default for this input is "No Action".

The switch connection is described in “Connecting Switches or Devices to EStop/Invert Input” on page 62. The switch must be such that it is in the open state in the normal situation and closed to signal an emergency stop command.

**After and Emergency Stop condition, the controller must be reset or powered Off and On to resume normal operation.**

---

## Inverted Operation

For robots that can run upside-down, the controller can be configured to reverse the motor commands using a gravity activated switch when the robot is flipped. This feature is enabled only in the mixed mode and when the switch is enabled with the proper configuration of the “Input switch function” parameter. See “Programmable Parameters List” on page 174.

The switch connection is described in “Connecting Switches or Devices to EStop/Invert Input” on page 62. The switch must be such that it is in the open state when the robot is in the normal position and closed when inverted. When the status of the switch has changed, the controller will wait until the new status has remained stable for 0.5s before acknowledging it and inverting the commands. This delay is to prevent switch activation triggered by hits and bounces which may cause the controller to erroneously invert the commands.

---

## Special Use of Accessory Digital Inputs

The AX2550 includes two general purpose digital inputs identified as Input E and Input F. When an Encoder Module is installed, input E is disabled. The location of these inputs on the DB15 connector can be found in the section “I/O List and Pin Assignment” on page 58, while the electrical signal needed to activate them is shown on “Connecting Switches or Devices to Input F” on page 61.

By default, these inputs are ignored by the controller. However, the AX2550 may be configured to cause either of the following actions:

- Activate the buffered Output C
- Turn Off/On the power MOSFET transistors

These alternate modes can only be selected using the Roborun Utility (see “Control Settings” on page 181. Each of these modes is detailed below.

### Using the Inputs to Activate the Buffered Output

When this setting is selected, the buffered Output C will be On when the Input line is pulled to Ground (0V). The Output will be Off when the Input is pulled high.

This function makes it possible to drive solenoids or other accessories up to 2A at 24V using a very low current switch, for example.

### Using the Inputs to turn Off/On the Power MOSFET transistors

When this setting is selected, the controller’s Power MOSFET transistors will be active, and the controller will be operating normally, only when the input is pulled to ground.

When the input is pulled high, all the power MOSFETs are turned Off so that the motors are effectively disconnected from the controller.

This function is typically used to create a “dead man switch” when the controller is driven using an analog joystick. The motors will be active only while the switch is depressed. If the switch is left off for any reason, the motors will be disconnected and allowed to free-wheel rather than coming to an abrupt stop.

---

## Self-Test Mode

The AX2550 incorporates a simple Self-Test mode that performs the following functions:

- Display the software revision number on the LED display
- Ramp each motor up and down in both directions
- Internal parameters on the serial port output

The Self Test mode can be conveniently initiated using only the controller’s switches so that no radio or computer is needed.

To enter the Self Test mode, press and hold the **Set** button while resetting or powering up the controller.

After a few seconds, the LEDs will display a sequence of two numerical digits and an optional letter separated by dashes as shown in the examples below.



FIGURE 25. Press and hold “Set” to display version number and enter self-test

After these digits are displayed, the controller will attempt to power the motors. Motor 1 will be ramped from stop to full speed forward to full speed reverse and back to stop. Then the same operation will repeat on motor 2. After both motors have completed their ramps, the software revision will be displayed again and the motors will be ramped again. This sequence will repeat itself indefinitely until the controller is powered off or reset.

While in the Self Test mode, the AX2550 will continuously send a string of characters on the RS232 output line. This string will contain 12 or 13 two-digits hexadecimal number representing the 13 following operating parameters.

- Captured R/C Command 1 and 2
- Power Applied to Controller’s output stage
- Values applied to Analog inputs 1 and 2
- Amps on channel 1 and 2
- Internal Heat Sink temperatures 1 and 2
- Main Battery voltage
- Internal 12V voltage

- Encoder Speed or Position if module is installed

The entire string is repeated every 208 or 224 milliseconds (depending whether the encoder module is present) with the latest internal parameter values. This information can be logged using the Roborun Utility (see "Viewing and Logging Data in Analog and R/C Modes" on page 192).

The string and data format is described in "Analog and R/C Modes Data Logging String Format" on page 166.

## **Important Warning**

**Disconnect the Motor Power (thick red wires) from the battery and power the controller by applying 12V on the Power Control input if you do not wish the motors to be activated while in Self Test mode.**

This could be the case if you only wish to use the Self Test sequence to read the software revision number or to monitor the joystick capture.

**SECTION 6**

# Connecting Sensors and Actuators to Input/Outputs

This section describes the various inputs and outputs and provides guidance on how to connect sensors, actuators or other accessories to them.

---

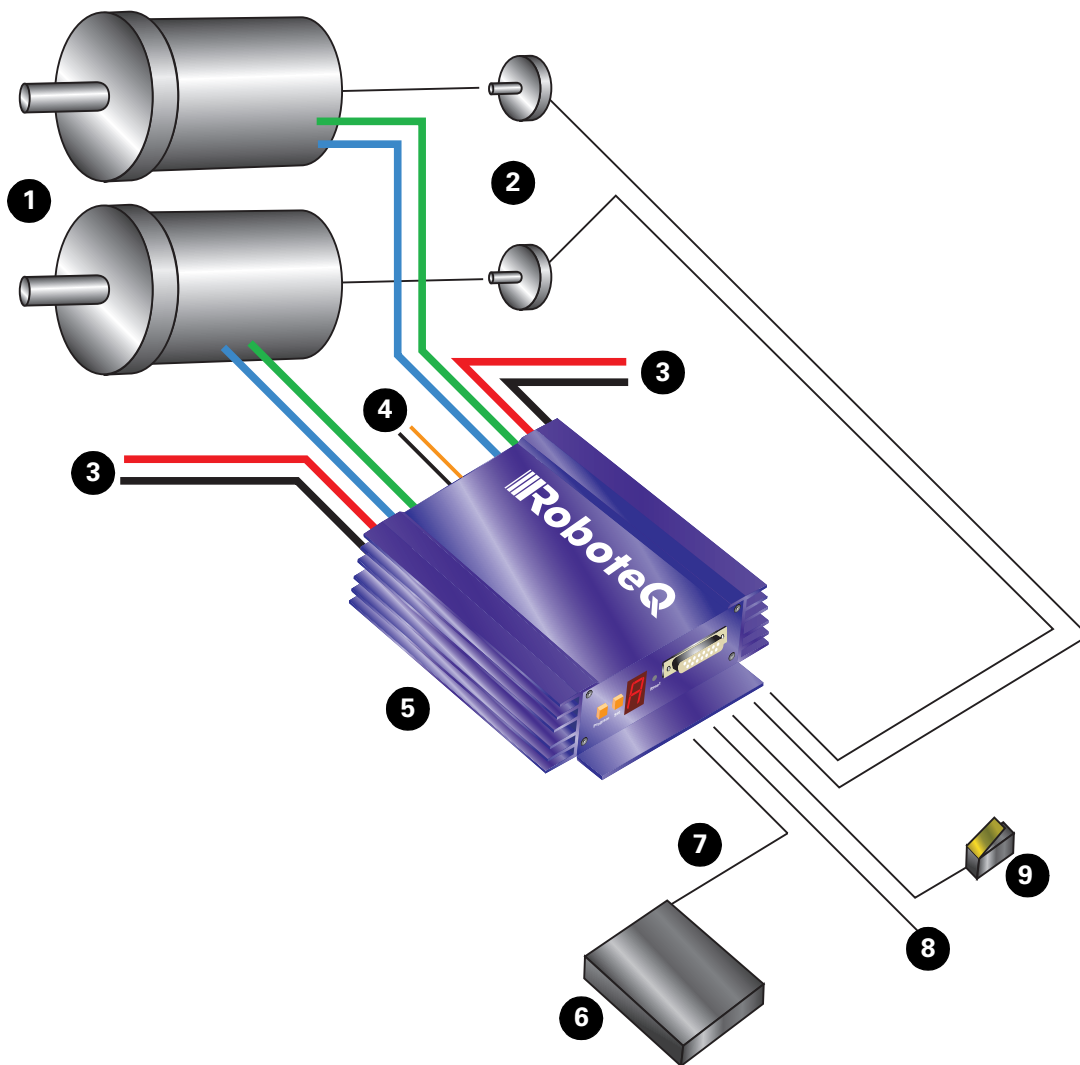
## **AX2550 Connections**

The AX2550 uses a set of power wires (located on the back of the unit) and a DB15 connector for all necessary connections. The diagram on the figure below shows a typical wiring diagram of a mobile robot using the AX2550 controller.

The wires are used for connection to the batteries and motors and will typically carry large current loads. Details on the controller's power wiring can be found at "Connecting Power and Motors to the Controller" on page 29

The DB15 connector is used for all low-voltage, low-current connections to the Radio, Microcontroller, sensors and accessories. This section covers only the connections to sensors and actuators.

For information on how to connect the R/C radio or the RS232 port, see "R/C Operation" on page 109 and "Serial (RS-232) Controls and Operation" on page 131.



- |  |   |
|--|---|
| 1- DC Motors   | 5- Controller   |
| 2- Optional sensors:<br>- Tachometers (Closed loop Speed mode)<br>- Potentiometers (Servo mode)<br>- Optical Encoder (AX2850 only - all closed loop modes) | 6- R/C Radio Receiver, microcomputer, or wireless modem |
| 3- Motor Power supply wires  | 7- Command: RS-232, R/C Pulse                           |
| 4- Power Control wire  | 8- Miscellaneous I/O                                    |
|  | 9- Running Inverted, or emergency stop switch           |

FIGURE 26. Typical controller connections



## AX2550's Inputs and Outputs

In addition to the RS232 and R/C channel communication lines, the AX2550 includes several inputs and outputs for various sensors and actuators. Depending on the selected operating mode, some of these I/Os provide feedback and/or safety information to the controller.

When the controller operates in modes that do not use these I/O, these signals become available for user application. Below is a summary of the available signals and the modes in which they are used by the controller or available to the user.

TABLE 9. AX2550 IO signals and definitions

Signal	I/O type	Use	Activated
Out C	2A Digital Output	User defined	Activated using R/C channel 3 (R/C mode), or serial command (RS232 mode) Activated when any one motor is powered (when enabled)
Inp F	Digital Input	User defined	Active in RS232 mode only. Read with serial command (RS232)
		Activate Output C	When Input is configured to drive Output C
		Turn FETs On/Off	When Input is configured as "dead man switch" input
Inp E	Digital Input	Same as Input F - (Not available when encoder module present)	
EStop/Invert	Digital Input	Emergency stop	When Input is configured as Emergency Stop switch input.
		Invert Controls	When Input is configured as Invert Controls switch input.
		User defined	When input is configured as general purpose. Read with serial command (RS232).
Analog In 1	Analog Input	Tachometers input	When Channel 1 is configured in Closed Loop Speed Control with Analog feedback
		Position sensing	When Channel 1 is configured in Closed Loop Position Control with RC or RS232 command and Analog feedback
		User defined	Read value with serial command (RS232).
Analog In 2	Analog Input 2	Same as Analog 1 but for Channel 2	
Analog In 3	Analog Input 3	Position sensing	When Channel 1 is configured in Closed Loop Position Control with Analog command and Analog feedback
		User defined	Read value with serial command (RS232).
Analog In 4	Analog Input 4	Same as Analog 3 but for Channel 4	

## I/O List and Pin Assignment

The figure and table below lists all the inputs and outputs that are available on the AX2550.

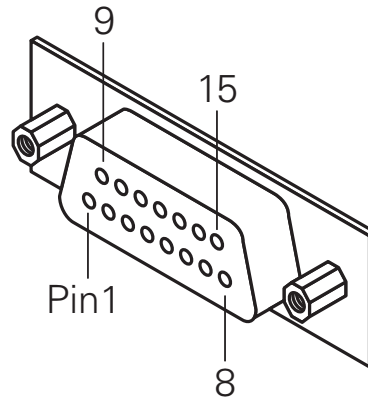


FIGURE 27. Controller’s DB15 connector pin numbering

TABLE 10. DB15 connector pin assignment

Pin Number	Input or Output	Signal depending on Mode	Description
1 and 9	Output	Output C	2A Accessory Output C
2	Output	R/C: Data Out	RS232 Data Logging Output
		RS232: Data Out	RS232 Data Out
		Analog: Data Out	RS232 Data Logging Output
3	Input	R/C: Ch 1	R/C radio Channel 1 pulses
		RS232: Data In	RS232 Data In (from PC/MCU)
		Analog: Unused	Unused
4	Input	R/C: Ch 2	R/C radio Channel 2 pulses
		RS232/Analog: Input F	Digital Input F readable RS232 mode Dead man switch activation
5 and 13	Power Out	Ground	Controller ground (-)
6	GND In Unused in RevB Hardware	Ground Unused in RevB Hardware	Optocoupler GND Input, Connect to pin 5** Unused in RevB Hardware

TABLE 10. DB15 connector pin assignment

Pin Number	Input or Output	Signal depending on Mode	Description
7	+5V In Unused in RevB Hardware	+5V Unused in RevB Hardware	Optocoupler +5V Input. Connect to pin 14** Unused in RevB Hardware
8	Digital In and Analog In	R/C: Ch 3	R/C radio Channel 3 pulses - (Not available on AX2850)
		RS232: Input E / Ana in 4	Accessory input E Dead man Switch Input Activate Output C Analog Input 4 in RevB Hardware
		Ana: Input E / Ana in 4	Accessory input E Dead man Switch Input Activate Output C Channel 2 speed or position feedback input in RevB Hardware
10	Analog in	RC/RS232: Ana in 2	Channel 2 speed or position feedback input
		Analog: Command 2	Analog command for channel 2
11	Analog in	RC/RS232: Ana in 1	Channel 1 speed or position feedback input
		Analog: Command 1	Analog command for channel 1
12		RC: Unused	
		RS232: Ana in 3	Analog input 3
		Ana: Ana in 3	Channel 1 speed or position feedback input in RevB Hardware
14	Power Out	+5V	+5V Power Output (100mA max.)
15	Input	Input EStop/Inv	Emergency Stop or Invert Switch input

\*\*These connections should only be done in RS232 mode or R/C mode with radio powered from the controller.

## Connecting devices to Output C

Output C is a buffered, Open Drain MOSFET output capable of driving over 2A at up to 24V.

The diagrams on Figure 28 show how to connect a light or a relay to this output:

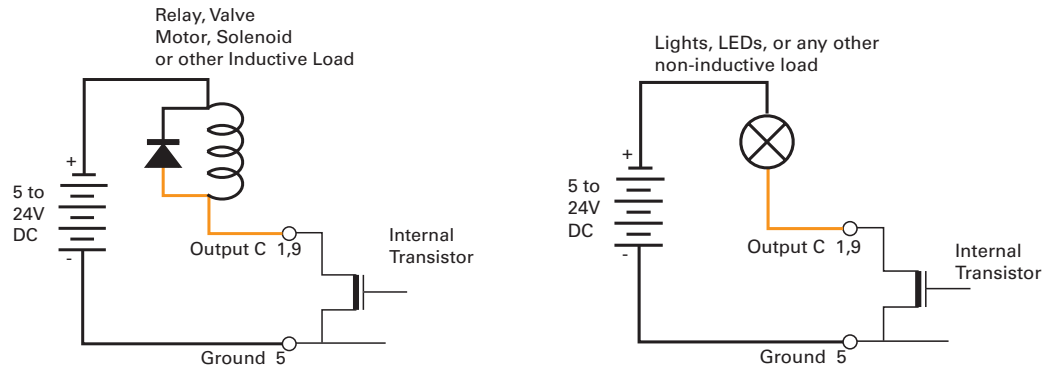


FIGURE 28. Connecting inductive and resistive loads to Output C

This output can be turned On and Off using the Channel 3 Joystick when in the R/C mode. See “Activating the Accessory Outputs” on page 120 for more information.

When the controller is used in RS232 mode, this output can be turned On and Off using the **!C** (On) and **!c** (Off) command strings. See “Controller Commands and Queries” on page 138 for more information.

### **Important warning:**

**Overvoltage spikes induced by switching inductive loads, such as solenoids or relays, will destroy the transistor unless a protection diode is used.**

### **Connecting Switches or Devices to Input E**

Input E is a general purpose, digital input. This input is only available if no encoder module is present and is active when in the RS232 and Analog modes. In R/C mode, this line is used as the radio channel 3 input.

Input E is a high impedance input with a pull-up resistor built into the controller. Therefore it will report an On state if unconnected, and a simple switch as shown on Figure 29 is necessary to activate it.

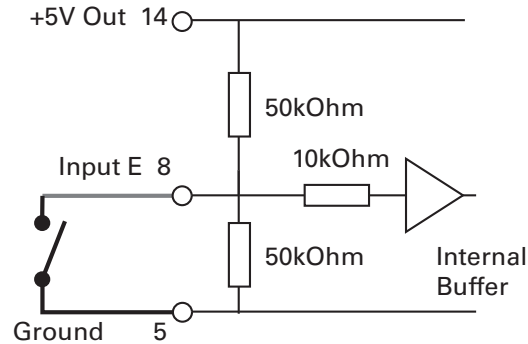


FIGURE 29. Switch wirings to Input E

The status of Input E can be read in the RS232 mode with the **?i** command string. The controller will respond with three sets of 2 digit numbers. The status of Input E is contained in the first set of numbers and may be 00 to indicate an Off state, or 01 to indicate an On state.

**This input is not available on the AX2850 or on the AX2550 with the Encoder Module installed.**

Remember that, in RevB hardware, InputE is shared with the Analog Input 4. If an analog sensor is connected, the controller will return a Digital value of 0 if the voltage is lower than 0.5V and a value of 1 if higher

## Connecting Switches or Devices to Input F

Input F is a general purpose digital input. This input is only active when in the RS232 or Analog modes. In R/C mode, this line is used as the radio channel 2 input.

When left open, Input F is in an undefined stage. As shown in the figure below, a pull down or pull up resistor must be inserted when used with a single pole switch. The resistor may be omitted when used with a dual pole switch.

On controllers prior to RevB, Input F is an opto-coupled input and requires the buffer to be powered with a connection between pin 14 and 7, and between 13 and 6, to operate. This power connection is not needed on RevB hardware.

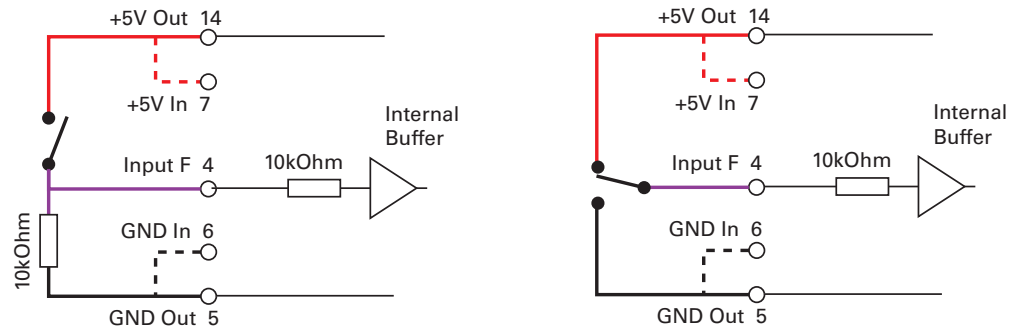


FIGURE 30. Switch wiring to Input F

The status of Input F can be read in the RS232 mode with the `?i` command string. The controller will respond with three sets of 2 digit numbers. The status of Input F is contained in the second set of numbers and may be 00 to indicate an Off state, or 01 to indicate an On state.

## Connecting Switches or Devices to EStop/Invert Input

This input is used to connect various switches or devices depending on the selected controller configuration.

The factory default for this input is "No Action."

This input can also be configured to be used with an optional "inverted" sensor switch. When activated, this will cause the controls to be inverted so that the robot may be driven upside-down.

When neither Emergency Stop or Inverted modes are selected, this input becomes a general purpose input like the other two described above.

This input is a high impedance input with a pull-up resistor built into the controller. Therefore it will report an On state (no emergency stop, or not inverted) if unconnected. A simple switch as shown on Figure 31 is necessary to activate it. Note that to trigger an Emergency Stop, or to detect robot inversion this input must be pulled to ground. Figure 31 show how to wire the switch to this input.

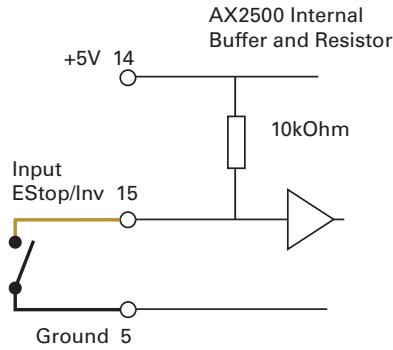


FIGURE 31. Emergency Stop / Invert switch wiring

The status of the EStop/Inv can be read at all times in the RS232 mode with the **?i** command string. The controller will respond with three sets of 2 digit numbers. The status of the ES/Inv Input is contained in the last set of numbers and may be 00 to indicate an Off state, or 01 to indicate an On state.

## Analog Inputs

The controller has (2 on pre-RevB hardware) 4 Analog Inputs that can be used to connect position, speed, temperature, voltage or most other types of analog sensors. These inputs can be read at any time using the **?p** query for Analog inputs 1 and 2 and the **?r** query for Inputs 3 and 4. The following section show the various uses for these inputs.

## Connecting Position Potentiometers to Analog Inputs

When configured in the Position mode, the controller's analog inputs are used to obtain position information from a potentiometer coupled to the motor axle. This feature is useful in order to create very powerful servos as proposed in the figure below:

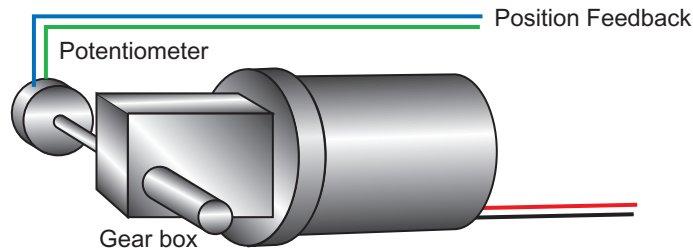


FIGURE 32. Motor and potentiometer assembly for position servo operation

Connecting the potentiometer to the controller is as simple as shown in the diagram on Figure 33.

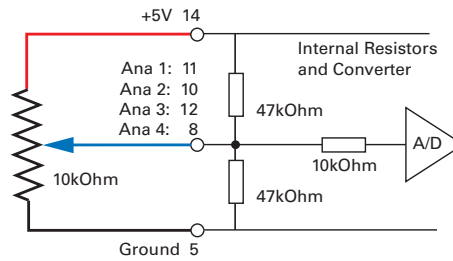


FIGURE 33. Potentiometer wiring in Position mode

The potentiometer must be attached to the motor frame so that its body does not move in relationship with the motor. The potentiometer axle must be firmly connected to the gearbox output shaft. The gearbox must be as tight as possible so that rotation of the motor translates into direct changes to the potentiometers, without slack, at the gearbox’s output.

TABLE 11. Analog Position Sensor connection depending on operating mode

Operating Mode	Ana 1 pin 11	Ana2 pin 10	Ana 3 pin 12	Ana 4 pin 8
RC or RS232 - Dual Channel	Position 1	Position 2	Unused	Unused
Analog - Dual Channel	Command 1	Command 2	Position 1	Position 2
RC or RS232 - Single Channel	Position	Unused	Unused	Unused
RC or RS232 - Dual Channel	Command	Unused	Position	Unused

See “Closed Loop Position Mode” on page 85 for complete details on Position Mode wiring and operation.

## Important Warning

**Beware that the wrong + and - polarity on the potentiometer will cause the motor to turn in the wrong direction and not stop. The best method to figure out the right potentiometer is try one way and change the polarity if incorrect. Note that while you are doing these tests, the potentiometer must be loosely attached to the motor’s axle so that it will not be forced and broken by the motor’s uncontrolled rotation in case it was wired wrong.**

## Connecting Tachometer to Analog Inputs

When operating in closed loop speed mode, tachometers must be connected to the controller to report the measured motor speed. The tachometer can be a good quality brushed DC motor used as a generator. The tachometer shaft must be directly tied to that of the motor with the least possible slack.



Since the controller only accepts a 0 to 5V positive voltage as its input, the circuit shown in Figure 34 must be used between the controller and the tachometer: a 10kOhm potentiometer is used to scale the tachometer output voltage to -2.5V (max reverse speed) and +2.5V (max forward speed). The two 1kOhm resistors form a voltage divider that sets the idle voltage at mid-point (2.5V), which is interpreted as the zero position by the controller. The voltage divider resistors should be of 1% tolerance or better. To precisely adjust the 2.5V midpoint value it is recommended to add a 100 ohm trimmer on the voltage divider.

With this circuitry, the controller will see 2.5V at its input when the tachometer is stopped, 0V when running in full reverse, and +5V in full forward.

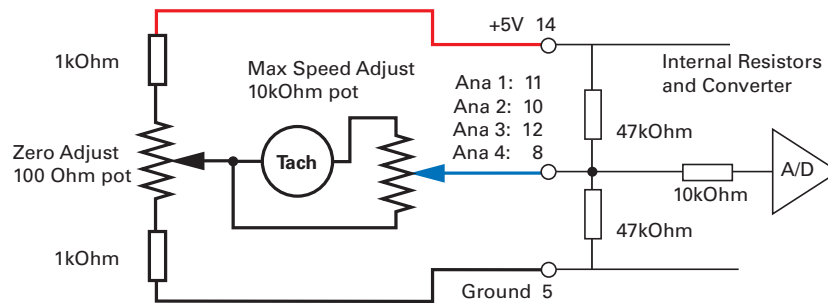


FIGURE 34. Tachometer wiring diagram

The tachometers can generate voltages in excess of 2.5 volts at full speed. It is important, therefore, to set the potentiometer to the minimum value (cursor all the way down per this drawing) during the first installation.

Since in closed loop control the measured speed is the basis for the controller's power output (i.e. deliver more power if slower than desired speed, less if higher), an adjustment and calibration phase is necessary. This procedure is described in "Closed Loop Speed Mode" on page 97.

TABLE 12. Analog Speed Sensor connection depending on operating mode

Operating Mode	Ana 1 (p11)	Ana2 (p10)	Ana 3 (p12)	Ana 4 (p8)
RC or RS232 - Dual Channel	Speed 1	Speed 2	Unused	Unused
Analog - Dual Channel	Command 1	Command 2	Speed 1	Speed 2
RC or RS232 - Single Channel	Speed	Unused	Unused	Unused
RC or RS232 - Dual Channel	Command	Unused	Speed	Unused

## Important Warning

**The tachometer's polarity must be such that a positive voltage is generated to the controller's input when the motor is rotating in the forward direction. If the polarity is inverted, this will cause the motor to run away to the maximum speed as soon as the controller is powered with no way of stopping it other than pressing the emergency stop button or disconnecting the power.**

## Connecting External Thermistor to Analog Inputs

Using external thermistors, the AX2550 can be made to supervise the motor's temperature and adjust the power output in case of overheating. Connecting thermistors is done according to the diagram show in Figure 35. The AX2550 is calibrated using a 10kOhm Negative Coefficient Thermistor (NTC) with the temperature/resistance characteristics shown in the table below.

TABLE 13. Recommended NTC characteristics

Temp (oC)	-25	0	25	50	75	100
Resistance (kOhm)	86.39	27.28	10.00	4.16	1.92	0.93

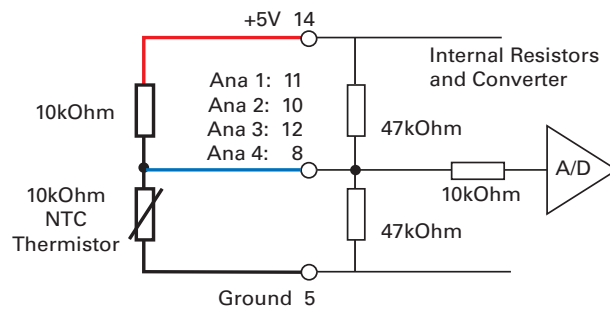


FIGURE 35. NTC Thermistor wiring diagram

Thermistors are non-linear devices. Using the circuit described on Figure 35, the controller will read the following values (represented in signed binary) according to the temperature.

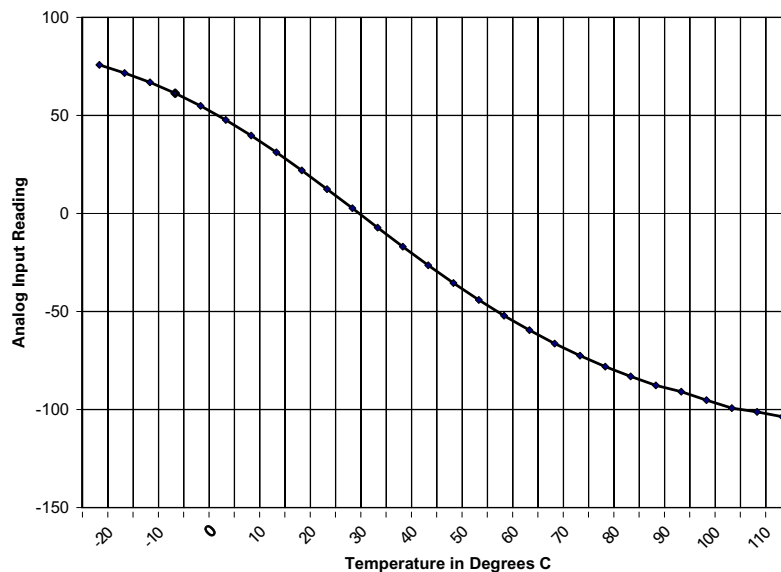


FIGURE 36. Signed binary reading by controller vs. NTC temperature

To read the temperature, use the **?p** command to have the controller return the A/D converter's value. The value is a signed 8-bit hexadecimal value. Use the chart data to convert the raw reading into a temperature value.

## Using the Analog Inputs to Monitor External Voltages

The analog inputs may also be used to monitor the battery level or any other DC voltage. In this mode, the controller does not use the voltage information but merely makes it available to the host microcomputer via the RS232 port. The recommended schematic is shown in Figure 37.

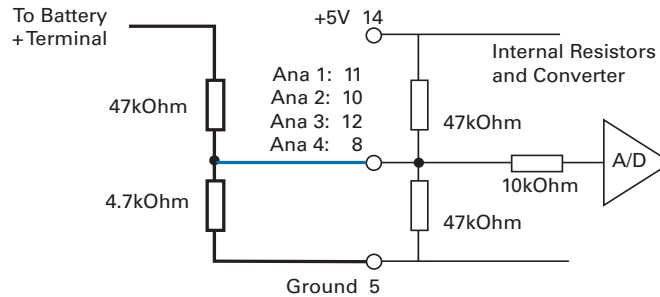


FIGURE 37. Battery voltage monitoring circuit

Using these resistor values, it is possible to measure a voltage ranging from -5V to +60V with a 0.25V resolution. The formula for converting the A/D reading into a voltage value is as follows.

$$\text{Measured volts} = ((\text{controller reading} + 128) * 0.255) - 5$$

Note: The A/D converter's reading is returned by the **?p** command and is a signed 8-bit hexadecimal value. You must add 128 to bring its range from -127/+127 to 0/255.

## Connecting User Devices to Analog Inputs

The two analog inputs can be used for any other purpose. The equivalent circuit for each input is shown in Figure 38. The converter operates with an 8-bit resolution, reporting a value of 0 at 0V and 255 at +5V. Care should be taken that the input voltage is always positive and does not exceed 5V. The converter's intrinsic diodes will clip any negative voltage or voltage above 5V, thus providing limited protection. The value of the analog inputs can be read through the controller's RS232 port.

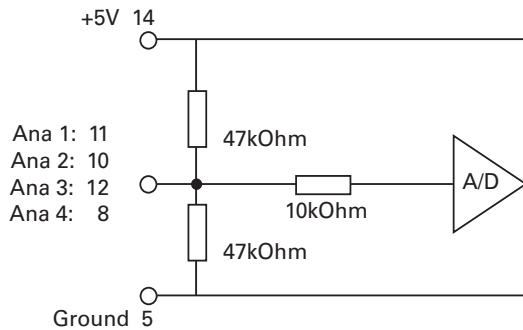


FIGURE 38. AX2550 Analog Input equivalent circuit

## Internal Voltage Monitoring Sensors

The AX2550 incorporates voltage sensors that monitor the Main Battery voltage and the Internal 12V supply. This information is used by the controller to protect it against overvoltage and undervoltage conditions (see “Overvoltage Protection” on page 38 and “Undervoltage Protection” on page 38). These voltages can also be read from the RS232 serial port using the **?e** query.

The returned value are numbers ranging from 0 to 255. To convert these numbers into a Voltage figure, the following formulas must be used:

$$\text{Measured Main Battery Volts} = 55 * \text{Read Value} / 256$$

$$\text{Measured Internal Volts} = 28.5 * \text{Read Value} / 256$$

## Internal Heatsink Temperature Sensors

The AX2550 includes temperature sensors making contact with the each of the two Heatsink sides of the controller.

These sensors are used to automatically reduce the maximum Amps that the controller can deliver as it overheats. However, the temperature can be read using the RS232 port using the **?m** query, or during data logging (see “Analog and R/C Modes Data Logging String Format” on page 166)

The analog value that is reported will range from 0 (warmest) to 255 (coldest). Because of the non-linear characteristics of NTC thermistors, the conversion from measured value to temperature must be done using the correction curve below.

It should be noted that the temperature is measured inside the controller and that it may be temporarily be different than the temperature measured outside the case.

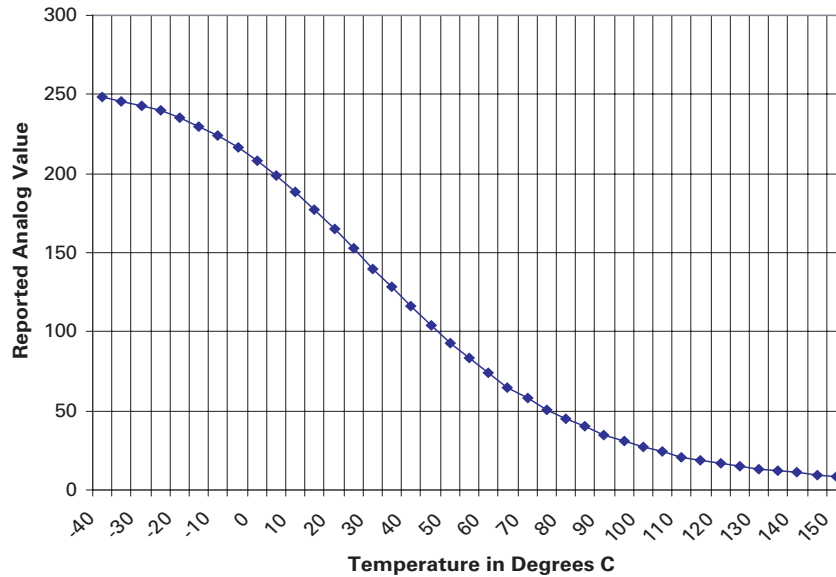


FIGURE 39. Analog reading by controller vs. internal heat sink temperature

## Temperature Conversion C Source Code

The code below can be used to convert the analog reading into temperature. It is provided for reference only. Interpolation table is for the internal thermistors.

```
int ValToHSTemp(int AnaValue)
{
    // Interpolation table. Analog readings at -40 to 150 oC, in 5o intervals

    int TempTable[39] = {248, 246, 243, 240, 235, 230, 224, 217, 208, 199, 188, 177,
        165, 153, 140, 128, 116, 104, 93, 83, 74, 65, 58, 51, 45, 40, 35, 31, 27, 24, 21,
        19, 17, 15, 13, 12, 11, 9, 8};

    int LoTemp, HiTemp, lobound, hibound, temp, i;

    i = 38;

    while (TempTable[i] < AnaValue && i > 0)
        i--;

    if (i < 0)
        i = 0;
    if (i == 38)
        return 150;
    else
    {
        LoTemp = i * 5 - 40;
        HiTemp = LoTemp + 5;
        lobound = TempTable[i];
        hibound = TempTable[i+1];
        temp = LoTemp + (5 * ((AnaValue - lobound)*100/ (hibound - lobound)))/100;
        return temp;
    }
}
```



## SECTION 7

# Installing, Connecting and Using the Encoder Module

This section describes the Encoder input module that is factory installed inside the AX2850 or that may be added onto the AX2550.

---

## Optical Incremental Encoders Overview

Optical incremental encoders are a means for capturing speed and travelled distance on a motor. Unlike absolute encoders which give out a multi-bit number (depending on the resolution), incremental encoders output pulses as they rotate. Counting the pulses tells the application how many revolutions, or fractions of, the motor has turned. Rotation velocity can be determined from the time interval between pulses or by the number of pulses within a given time period. Because they are digital devices, incremental encoders will measure distance and speed with perfect accuracy.

Since motors can move in forward and reverse directions, it is necessary to differentiate the manner that pulses are counted so that they can increment or decrement a position counter in the application. Quadrature encoders have dual channels, A and B, which are electrically phased 90° apart. Thus, direction of rotation can be determined by monitoring the phase relationship between the two channels. In addition, with a dual-channel encoder, a four-time multiplication of resolution can be achieved by counting the rising and falling edges of each channel (A and B). For example, an encoder that produces 250 Pulses per Revolution (PPR) can generate 1,000 Counts per Revolution (CPR) after quadrature.

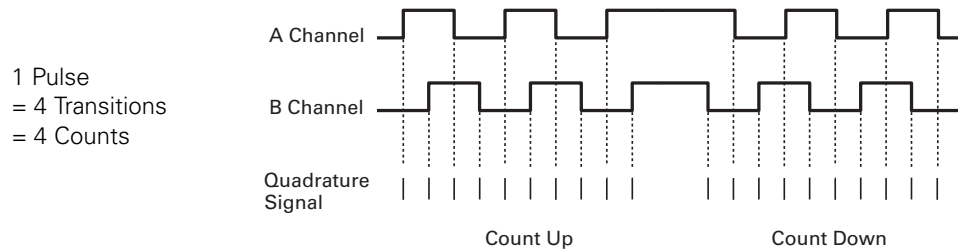


FIGURE 40. Quadrature encoder output waveform

The figure below shows the typical construction of a quadrature encoder. As the disk rotates in front of the stationary mask, it shutters light from the LED. The light that passes through the mask is received by the photo detectors. Two photo detectors are placed side by side so that the light making it through the mask hits one detector after the other to produce the 90° phased pulses.

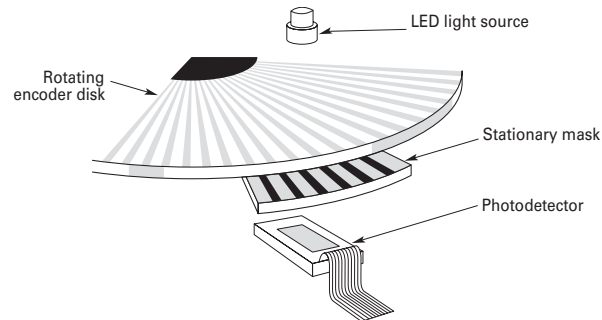


FIGURE 41. Typical quadrature encoder construction

Unlike absolute encoders, incremental encoders have no retention of absolute position upon power loss. When used in positioning applications, the controller must move the motor until a limit switch is reached. This position is then used as the zero reference for all subsequent moves.

## Recommended Encoder Types

The module may be used with most incremental encoder module as long as they include the following features:

- Two quadrature outputs (Ch A, Ch B), single ended signal
- 2.5V minimum swing between 0 Level and 1 Level on quadrature output
- 5VDC operation. 100mA or less current consumption per encoder

More sophisticated incremental encoders with differential outputs, index, and other features may be used, however these additional capabilities will be ignored.

The choice of encoder resolution is very wide and is constrained by the module's maximum pulse count at the high end and measurement resolution for speed at the low end.



Specifically, the encoder module can process 250,000 counts per seconds. As discussed in the previous section, a count is generated for each transition on the Channel A and Channel B. Therefore the module will work with encoders outputting up to 62,500 pulses per second.

Commercial encoders are rated by their numbers of “Pulses per Revolution” (also sometimes referred as “Cycles per Revolution”). Carefully read the manufacturer’s datasheet to understand whether this number represents the number of pulses that are output by each channel during the course of a 360o revolution rather than the total number of transitions on both channels during a 360o revolution. The second number is 4 times larger than the first one.

The formula below gives the pulse frequency at a given RPM and encoder resolution in Pulses per Revolution.

$$\text{Pulse Frequency in Hz} = \text{RPM} / 60 * \text{PPR} * 4$$

Example: a motor spinning at 10,000 RPM max, with an encoder with 200 Pulses per Revolution would generate:

$10,000 / 60 * 200 * 4 = 133.3 \text{ kHz}$  which is well within the 250kHz maximum supported by the encoder module.

An encoder with a 200 Pulses per Revolutions is a good choice for most applications.

A higher resolution will cause the counter to count faster than necessary and possibly reach the encoder module’s maximum frequency limit.

An encoder with a much lower resolution will cause speed to be measured with less precision.

## Installing the Encoder Module

The Encoder module is available in kit form for installation by the user inside the AX2550 controller. It may also be ordered factory-installed in the controller by using the product reference AX2850.

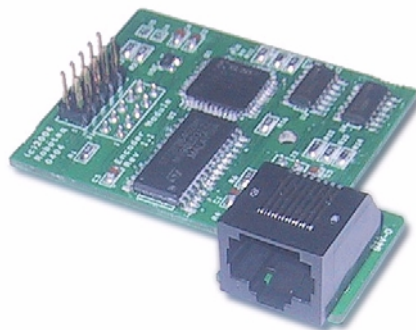


FIGURE 42. Encoder Module for AX2550. Factory installed in AX2850

Follow these simple steps to install:

- 1- With the power removed, remove the controller's face plate and slide off the cover.
- 2- Carefully insert the encoder module on top of the two headers present on the controller's main board and shown in Figure 43. Beware that the two matting connectors are precisely aligned.

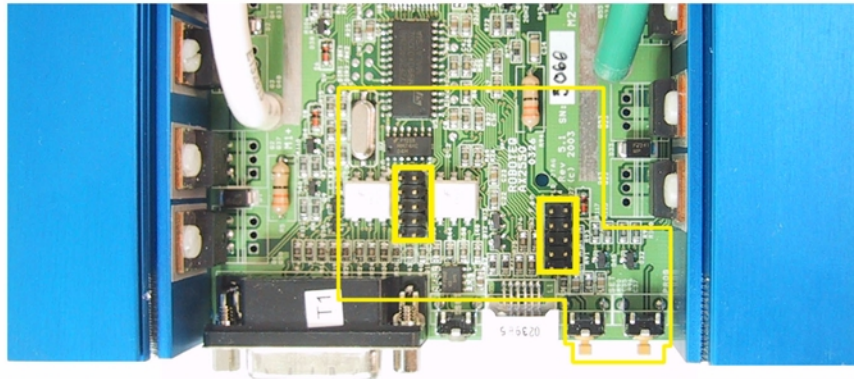


FIGURE 43. Position of Encoder Module on Controller's main board

- 3- The encoder module will be held in place by the connectors and the new front bracket's opening around the RJ11 connector. For added stability, place the foam pad on top of the Encoder module as shown in Figure 44 so that it is held in place by the sliding cover. For use in harsh shock and vibration environments, you may instead solder a metal wire inside the 0.1" hole found on the main board (next to one of the two header) and solder the other end inside the matching hole on the encoder module, as shown on Figure 45.

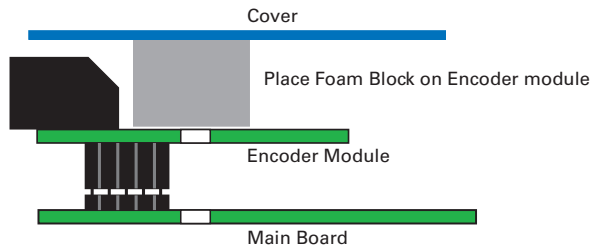


FIGURE 44. Place foam block between module and cover to hold in place

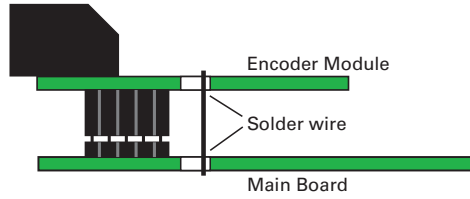


FIGURE 45. Solder wire for robust assembly

4- Slide the cover back on. Install the new face plate.

### Connecting the Encoder

The Encoder module uses a widely available 8-pin RJ45 connector identical to those found on all Ethernet devices. The connector provides 5V power to the encoders and has inputs for the two quadrature signals from each encoder. Using multi-level signaling, it is also possible to share the quadrature inputs with limit switches. The figure and table below describe the connector and its pin assignment.

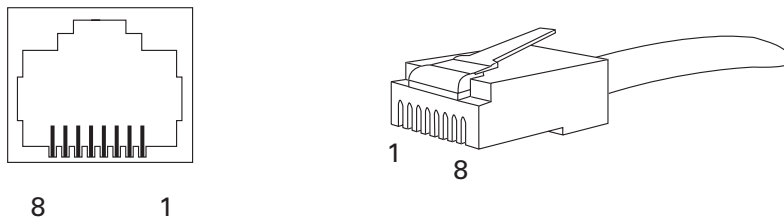


FIGURE 46. Encoder connector

TABLE 14. Encoder Connector Pinout

Pin	Name	Cable Color (when using standard network cable)
1	Encoder 2 - Channel B. Optional Limit Switch 4	Orange/White
2	Encoder 2 - Channel A. Optional Limit Switch 3	Orange
3	Ground (same as pin 7)	Green/White
4	5V Out (same as pin 8)	Blue
5	Encoder 1 - Channel B. Optional Limit Switch 2	Blue/White
6	Encoder 1 - Channel A. Optional Limit Switch 1	Green
7	Ground (same as pin 3)	Brown/White
8	5V Out (same as pin 4)	Brown

---

## Cable Length and Noise Considerations

Cable should not exceed one 3' (one meter) to avoid electrical noise to be captured by the wiring. A ferrite core filter must be used for length beyond 2' (60 cm). For longer cable length use an oscilloscope to verify signal integrity on each of the pulse channels and on the power supply.

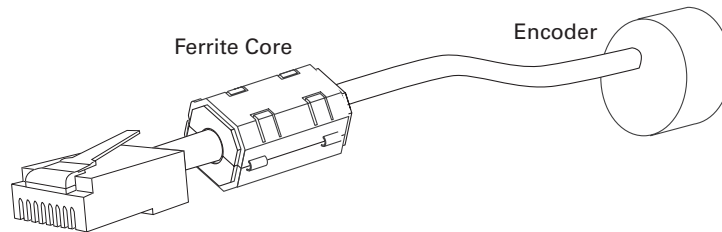


FIGURE 47. Use ferrite core on cable length beyond 2' or 60cm

---

## Important Warning

**Excessive cable length will cause electrical noise to be captured by the controller and cause erratic functioning that may lead to failure. In such situation, stop operation immediately.**

---

## Motor - Encoder Polarity Matching

When using the Encoder module for closed loop speed control, it is imperative that when the motor is turning in the forward direction, the counter increments its value and a positive speed value is measured.

Using the PC utility, it is possible to exercise the motors and view the encoder readings. See "Encoder Testing and Setting Using the PC Utility" on page 83.

If the Encoder counts backwards when the motor moves forward, correct this by either:

1- Swapping Channel A and Channel B on the encoder connector. This will cause the encoder module to reverse the count direction, or

2- Swapping the leads on the motor. This will cause the motor to rotate in the opposite direction.

---

## Voltage Levels, Thresholds and Limit Switches

The encoder module's input uses a comparator to reshape the encoder's output signal. If the signal is below a programmable threshold level, then it is considered to be 0. If above, it is considered to be 1. The output of this comparator feeds the quadrature detector and counters.

On the 2850 family of controllers, the threshold voltage may be changed under software control to any value between 0 and 5V in order to meet unusual encoder specifications. By default, the threshold level is 2.5V.

Another set of comparators on the same input signals detects pulses that are above and below a fixed 0.5V threshold. Using a special circuitry for creating multi-level signaling (see next section below), the output of these comparators serves to detect the status of optional limit switches.

Figure 49 and Figure 48 show the conditioned signals as seen by the encoder.

In Figure 49, the encoders are connected directly to the Channel A and B inputs. In this case, it will cause a Switch Detection condition because the encoder's 0 level is below 0.5V, which should be ignored.

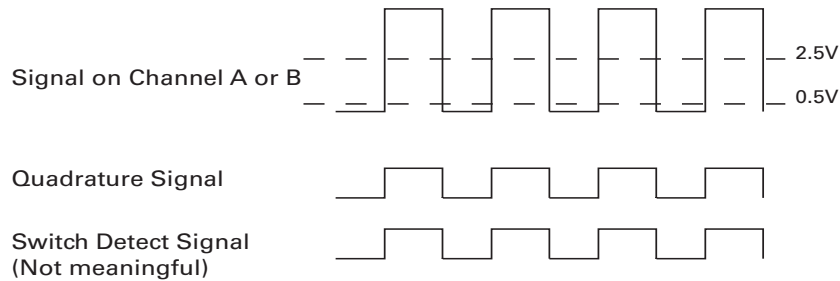


FIGURE 48. Signals seen by encoder using direct connection and no limit switches

In Figure 49, the encoder and switches are wired to the encoder module using a set of resistors designed to create a multi-level signal combining both pieces of information. Details on the necessary wiring is provided in the next section.

Since the encoder output signal is “shifted-up” by a few volts, it always stays above the Limit Switch comparator’s threshold, and no Switch Detection condition is generated. However, since the limit switches connect to ground when On, the level will dip below the 0.5V and generate a Switch Detection condition.

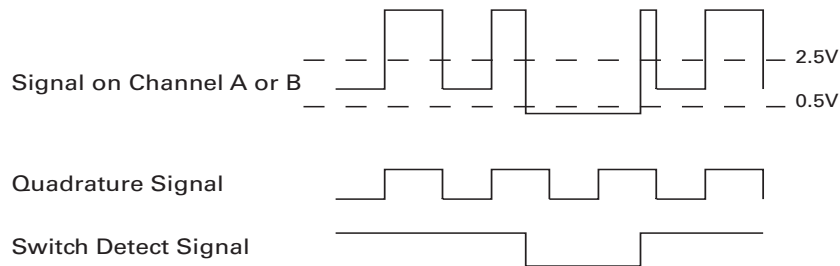


FIGURE 49. Signals seen by encoder using multi-levels and limit switches

## Important Warning

When a limit switch is activated, the encoder signal that is shared with the switch is no longer visible by the encoder module, and pulse counting and speed measurement stops.

## Wiring Optional Limit Switches

If limit switches are needed by the application, additional circuitry is required in order to create a multi-level signal that shares the encoder and the switch information. The figure below shows the electrical diagram of the required wiring.

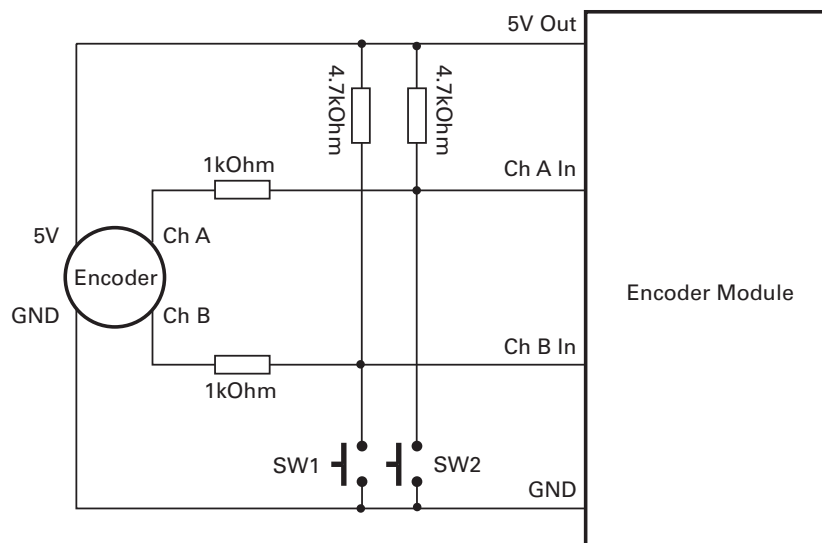


FIGURE 50. Signals seen by encoder using multi-levels and limit switches

Using this circuit when the switch is open, a 0V (low-level) output from the encoder goes through a 1k and 4.7k voltage divider, thus creating a voltage that will never be below 0.8V at the encoder module's input.

When the switch is activated, the module's input is pulled to 0V.

It is recommended that a voltmeter and/or oscilloscope be used to verify that the right voltage levels are created as the encoder rotates and the switches activate.

You may also use the Encoder setup/test function in the Roborun utility (see "Encoder Testing and Setting Using the PC Utility" on page 83). If the wiring is correct, the counters should increment/decrement as the motor rotate. The switch indicators should be always off unless the switches are actually activated.

## Wiring Limit Switches Without Encoders

If no encoder is used, the Encoder Module's inputs can be used to wire limit switches directly with solely a pull-up resistor as shown in the diagram below.

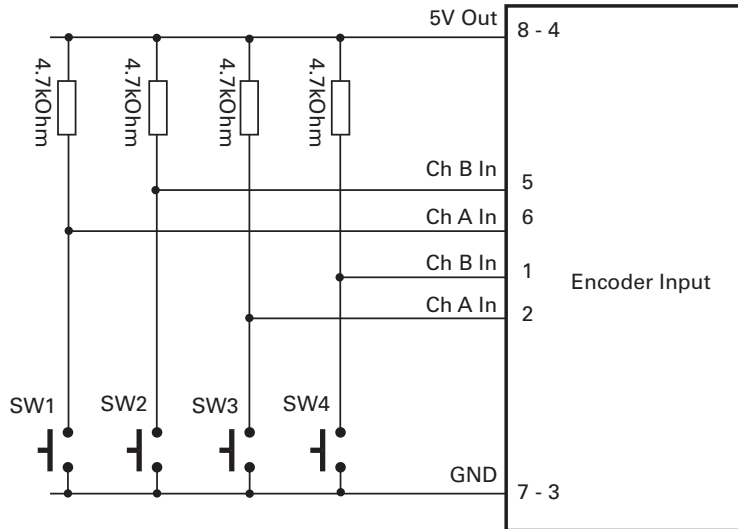


FIGURE 51. Signals seen by encoder using multi-levels and limit switches

## Effect of Limit Switches

Each pair of limit switches will stop the motion of a given motor in a given direction. This will have the effect of stopping the motor when a limit is reached while allowing motion in the other direction, away for that limit.

TABLE 15. Effects of Limit Switches 1 and 2 on Motor 1

SW1	SW2	Motor 1 Fwd	Motor 1 Rev
OFF	OFF	Allowed	Allowed
ON	OFF	Stopped	Allowed
OFF	ON	Allowed	Stopped
ON	ON	Stopped	Stopped

TABLE 16. Effects of Limit Switches 3 and 4 on Motor 2

SW3	SW4	Motor 2 Fwd	Motor 2 Rev
OFF	OFF	Allowed	Allowed
ON	OFF	Stopped	Allowed

TABLE 16. Effects of Limit Switches 3 and 4 on Motor 2

SW3	SW4	Motor 2 Fwd	Motor 2 Rev
OFF	ON	Allowed	Stopped
ON	ON	Stopped	Stopped

In Single Channel Mode, limit switches 3 and 4 are used. This is to allow direct connection of an encoder on the inputs for channel 1 and direct connection of switches on the inputs for channel 2

TABLE 17. Effects of Limit Switches 3 and 4 on Motor 2 in Single Channel Configuration

SW3	SW4	Motor Fwd	Motor Rev
OFF	OFF	Allowed	Allowed
ON	OFF	Stopped	Allowed
OFF	ON	Allowed	Stopped
ON	ON	Stopped	Stopped

## Using the Encoder Module to Measure Distance

As the encoders rotate, their quadrature outputs is automatically processed and increments/decrements two 32-bit counter inside the Encoder Module. There is one 32-bit counter for each of the encoders.

The counter values are stored as a signed binary numbers, ranging from -2,147,836,648 to +2,147,836,647 (Hexadecimal format of value 80000000 to 7FFFFFFF respectively. When the maximum or minimum counter values are reached, the counters automatically roll over to zero.

The counters can be read and set using the commands described in "The Encoder Module contains its own Microcontroller and firmware in Flash. When present, it responds to a large set of dedicated commands and queries via the controller's serial port. See "RS232 Encoder Command Set" on page 155." on page 82.

## Using the Encoder to Measure Speed

The encoder module will automatically compute rotation speed for each encoder. The resulting measured speed is a value ranging from 0 to + 127 and 0 to -127, where 127 represent a relative ratio of a maximum speed value chosen by the user.

For example, if the encoder module is configured so that the highest measured speed value is 3,000 RPM, then a reading of 63 (127/2) would be 1,500 RPM.

The relationship between the measured speed and the actual speed is a factor of two variable parameters: a Time Based period value stored inside the Encoder module and the Encoder's number of Pulses per Revolution. Note: the Encoder's number of Pulses per Revolution **is not stored** in the controller.

The Time Base is a number of 256us time intervals between two counter reads.



A simple procedure is included in the Roborun PC utility to easily determine and set these parameters.

For information, the exact formula is shown below:

$$\text{Measured Speed Value} = \text{RPM} * \text{PPR} * 4 * (\text{Time Base} + 1) * 256 / (60 * 1000000)$$

$$\text{or Measured Speed Value} = \text{RPM} * \text{PPR} * (\text{Time Base} + 1) / 58593.75$$

Example: a motor spinning at 1,000 RPM, with an encoder with 200 Pulses per Revolution, and a Time Base set at 4 will produce the following measurement:

$$1000 * 200 * (4 + 1) / 58593.75 = 17$$

The same formula modified to show the actual RPM at a given Measure Speed Value is as follows:

$$\text{RPM} = \text{Measured Speed Value} * 60 * 1000000 / (\text{PPR} * 4 * 256 * (\text{Time Base} + 1))$$

$$\text{or RPM} = \text{Measured Speed Value} * 58593.75 / ((\text{Time Base} + 1) * \text{PPR})$$

In our example, a measured speed value of 127 corresponds to the following measurable max actual RPM values.

$$\text{RPM at Max Measurable Speed Value} = 127 * 58593.75 / ((4 + 1) * 200) = 7441 \text{ RPM}$$

A measured speed value of 1 corresponds to the following measurable min. actual RPM values.

$$\text{RPM at Min. Measurable Speed Value} = 1 * 58593.75 / ((4 + 1) * 200) = 58.6 \text{ RPM}$$

The Roborun Utility automatically makes the above calculations when setting up the encoder.

## Important Notice

**The time base value should not exceed 63 so that a new speed value can be measured at every 16ms loop. The roborun utility automatically limits the time base value that can be entered.**

---

## Using the Encoder to Track Position

The encoder module can be used to report the distance between the actual motor position and a desired destination. The resulting measured "distance" can then be used by the controller in the position mode to move the motor in the right direction until the destination is reached. This movement is controlled by the PID position algorithm inside the controller and is therefore best suited at tracking position.

Since the controller uses a signed 8-bit value (-127 to +127) for the distance measurement in the Position Mode, a special algorithm is used to convert the real distance which can be much higher than -127 to +127, as both the counter and destination registers are 32-bit wide.

The actual formula is as follows:

$$\text{Distance} = (\text{Destination} - \text{Counter value}) / \text{Divider}$$

Where: divider is a configurable parameter of value 1, 2, 4, 8, 16, 32, 64 or 127

If computed distance is less than -127, then reported distance is -127

If computed distance is larger than +127, then reported distance is +127

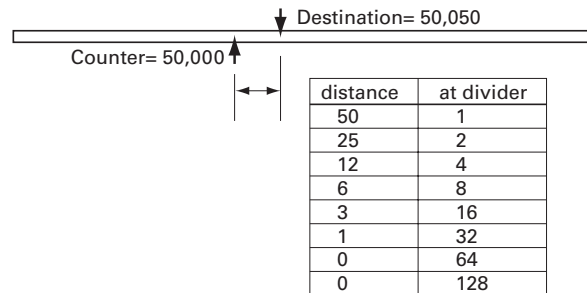


FIGURE 52. Small distance computation example

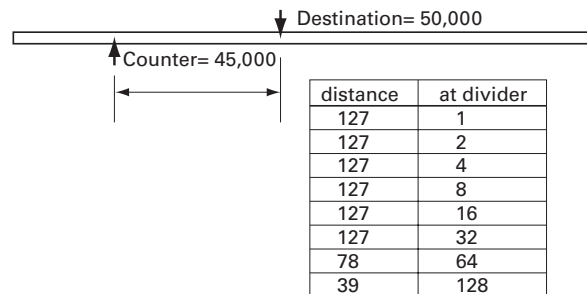


FIGURE 53. Large distance computation example

## Important Notice

**Encoders do not report an absolute position value but a count that is relative to the point where the motor shaft was at power up. It is typically necessary to have the motors moved to a "home" position and reset the counters at that reference point.**

## RS232 Communication with the Encoder Module

The Encoder Module contains its own Microcontroller and firmware in Flash. When present, it responds to a large set of dedicated commands and queries via the controller's serial port. See "RS232 Encoder Command Set" on page 155.

## **Encoder Testing and Setting Using the PC Utility**

Extensive diagnostic, calibration, setting and testing support is provided in the Roborun PC utility. Basic instructions on how to install and run the PC utility can be found in "Encoder Setting and Testing" on page 184.



## SECTION 8

# Closed Loop Position Mode

This section describes the AX2550 Position mode, how to wire the motor and position sensor assembly and how to tune and operate the controller in this mode.

---

## Mode Description

In this mode, the axle of a geared-down motor is coupled to a position sensor that is used to compare the angular position of the axle versus a desired position. The controller will move the motor so that it reaches this position.

This unique feature makes it possible to build ultra-high torque “jumbo servos” that can be used to drive steering columns, robotic arms, life-size models and other heavy loads.

The AX2550 incorporates a full-featured Proportional, Integral, Differential (PID) control algorithm for quick and stable positioning.

---

## Selecting the Position Mode

The position mode is selected by changing the Motor Control parameter in the controller to either

- A Open Loop Speed, B Position
- A Closed Loop Speed, B Position
- A and B Position

Note that in the first two modes, only the second motor will operate in the Position mode.

Changing the parameter is best done using the Roborun Utility. See “Loading, Changing Controller Parameters” on page 181.

For safety reasons and to prevent this mode from being accidentally selected, Position modes **CANNOT** be selected by configuring the controller using the built-in switches and display.

---

## Position Sensor Selection

The AX2550 may be used with the following kind of sensors:

- Potentiometers
- Hall effect angular sensors
- Optical Encoders (with Encoder Module)

The first two are used to generate an analog voltage ranging from 0V to 5V depending on their position. They will report an absolute position information at all times.

Optical encoders report incremental changes from a reference which is their initial position when the controller is powered up or reset. Using Optical Encoders in this mode is possible but requires special handling that is described in Figure , "Using the Encoder to Track Position," on page 81.

---

## Sensor Mounting

Proper mounting of the sensor is critical for an effective and accurate position mode operation. Figure 54 shows a typical motor, gear box, and sensor assembly.

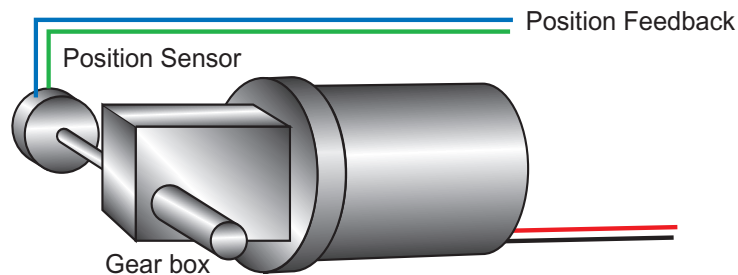


FIGURE 54. Typical motor/potentiometer assembly in Position Mode

---

The sensor is composed of two parts:

- a body which must be physically attached to a non-moving part of the motor assembly or the robot chassis, and
- an axle which must be physically connected to the rotating part of the motor you wish to position.

A gear box is necessary to greatly increase the torque of the assembly. It is also necessary to slow down the motion so that the controller has the time to perform the position control algorithm. If the gearing ratio is too high, however, the positioning mode will be very sluggish.

A good ratio should be such that the output shaft rotates at 1 to 10 rotations per second (60 to 600 RPM) when the motor is at full speed.

The mechanical coupling between the motor and the sensor must be as tight as possible. If the gear box is loose, the positioning will not be accurate and will be unstable, potentially causing the motor to oscillate.

Some sensor, such as potentiometers, have a limited rotation range of typically 270 degrees (3/4 of a turn), which will in turn limit the mechanical motion of the motor/potentiometer assembly. Consider using a multi-turn potentiometer as long as it is mounted in a manner that will allow it to turn throughout much of its range, when the mechanical assembly travels from the minimum to maximum position.

## **Important Notice:**

**Potentiometers are mechanical devices subject to wear. Use better quality potentiometers and make sure that they are protected from the elements. Consider using a solid state hall position sensor in the most critical applications. Optical encoders may also be used when operated as discussed in "Using the Encoder to Measure Speed" on page 80.**

---

## **Feedback Potentiometer wiring**

When using a potentiometer, it must be wired so that it creates a voltage that is proportional to its angular position: 0V at one extreme, +5V at the other. A 10K potentiometer value is recommended for this use.

Analog Feedback is normally connected to the Analog Inputs 1 and 2, except when the controller is configured in Analog Mode. In Analog mode, Analog Inputs 1 and 2 are already used to supply the command. Therefore Analog inputs 3 and 4 are used for feedback

## **Feedback Potentiometer wiring in RC or RS232 Mode**

In RC or RS232 mode, feedback is connected to Analog Inputs 1 and 2. Connecting the potentiometer to the controller is as simple as shown in the diagram on below.

Note that this wiring must not be used if the controller is configured in Analog mode but is switched in RS232 after power up using the method discussed in "Entering RS232 from R/C or Analog mode" on page 136. Instead, used the wiring for Analog mode discussed in the next section.

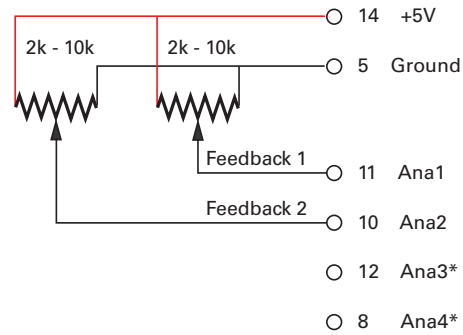


FIGURE 55. Pot wiring for RS232 or RC Command and Analog Feedback

### Feedback Potentiometer wiring in Analog Mode

When the controller is configured in Analog mode, the analog inputs 1 and 2 are used for commands while the analog inputs 3 and 4 are used for feedback. Analog inputs 3 and 4 have different characteristics than inputs 1 and 2, and so require a lower resistance potentiometer in order to guarantee accuracy

Note that the analog inputs 3 and 4 are only available on the AX2550 with PCB revision 6.1 or above. The PCB revision can be found on the sticker affixed at the bottom of the case. Roborun will detect the new hardware revision and display **Rev B** on the screen.

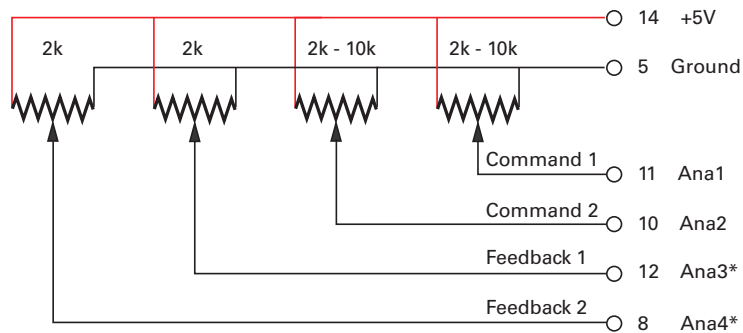


FIGURE 56. Pot wiring for Analog Command and Analog Feedback

Analog inputs 3 and 4 have different characteristics than inputs 1 and 2, and so require a lower resistance potentiometer in order to guarantee accuracy.

### Important Notice

This wiring is also the one to use when the controller is in Analog mode but switched to RS232 after reset using the method discussed in "Entering RS232 from R/C or Analog mode" on page 136



## Analog Feedback on Single Channel Controllers

On Single Channel controllers (SC Version - not to be confused with Dual Channel controllers of which only one channel is used for position control - See "Single Channel Operation" on page 177.), the controller accepts one command and uses one input for feedback.

### Feedback Wiring in RC or RS232 Mode on Single Channel Controllers

When the controller is configured for RS232 or RC command, the wiring of the feedback must be done as shown in the figure below.

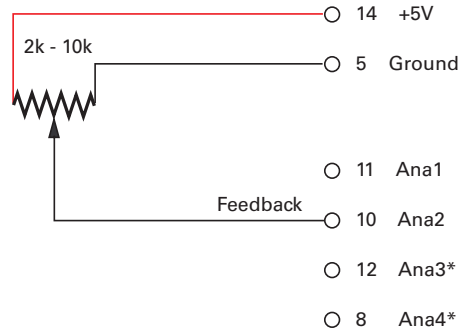


FIGURE 57. Pot wiring on Single Channel controllers (SCversion) and Analog Command

### Feedback Wiring in Analog Mode on Single Channel Controllers

When the controller is configured in Analog mode, the analog input 1 is used for commands while the analog input 4 is used for feedback. Note that the analog inputs 3 and 4 are only available on the AX2550 with PCB revision 6.1 or above. The PCB revision can be found on the sticker affixed at the bottom of the case. Roborun will detect the new hardware revision and display **Rev B** on the screen.

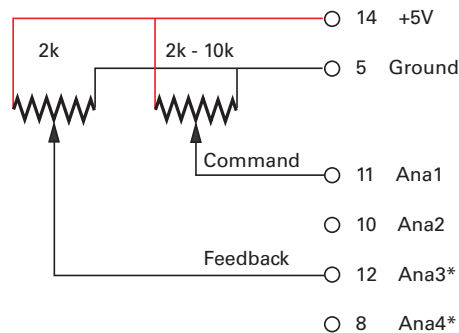


FIGURE 58. Pot wiring on Single Channel controllers (SC version) and Analog Command

Analog inputs 3 and 4 have different characteristics than inputs 1 and 2, and so require a lower resistance potentiometer in order to guarantee accuracy.

## Important Notice

This wiring is also the one to use when the controller is in Analog mode but switched to RS232 after reset using the method discussed in “Entering RS232 from R/C or Analog mode” on page 136

---

## Using Optical Encoders in Position Mode

The AX2550 and AX1500 may be equipped with an optional Optical Encoder Module. Optical Encoders require special handling. See Figure 7, “Installing, Connecting and Using the Encoder Module,” on page 71 for a detailed discussion.

---

## Sensor and Motor Polarity

The sensor polarity (i.e. which rotation end produces 0 or 5V) is related to the motor’s polarity (i.e. which direction the motor turns when power is applied to it).

In the Position mode, the controller compares the actual position, as measured by the sensor, to the desired position. If the motor is not at that position, the controller will apply power to the motor so that it turns towards that destination until reached.

## Important Warning:

**If there is a polarity mismatch, the motor will turn in the wrong direction and the position will never be reached. The motor will turn continuously with no way of stopping it other than cutting the power or hitting the Emergency Stop button.**

Determining the right polarity is best done experimentally using the Roborun utility (see “Using the Roborun Configuration Utility” on page 177) and following these steps:

1. Disconnect the controller’s Motor Power (thick power wires).
2. Configure the controller in Position Mode using the PC utility.
3. Loosen the sensor’s axle from the motor assembly.
4. Launch the Roborun utility and click on the Run tab. Click the “Start” button to begin communication with the controller. The sensor values will be displayed in the Ana1 and Ana2 boxes.
5. Move the sensor manually to the middle position until a value of “0” is measured using Roborun utility
6. Verify that the motor sliders are in the “0” (Stop) position. Since the desired position is 0 and the measured position is 0, the controller will not attempt to move the motors. The Power graph on the PC must be 0.
7. Apply power to the Motor Power input (thick power wires). The motor will be stopped.
8. With a hand ready to disconnect the Motor Power cable or ready to press the “Program” and “Set” buttons at the same time (Emergency Stop), SLOWLY move the sensor off the center position and observe the motor’s direction of rotation.

9. If the motor turns in the direction in which the sensor was moved, the polarity is correct. The sensor axle may be tighten to the motor assembly.
10. If the motor turns in the direction away from the sensor, then the polarity is reversed. The wire polarity on the motors should be exchanged. If using a potentiometer as sensor, the GND and +5V wires on the potentiometer may be swapped instead. If using an Optical Encoder, ChA and ChB outputs can be swapped.
11. Move the sensor back to the center point to stop the motor. Cut the power if control is lost.
12. If the polarity was wrong, invert it and repeat steps 8 to 11.
13. Tighten the sensor.

## **Important Safety Warning**

**Never apply a command that is lower than the sensor's minimum output value or higher than the sensor's maximum output value as the motor would turn forever trying to reach a position it cannot. For example, if the max position of a potentiometer is 4.5V, which is a position value of 114, a destination command of 115 cannot be reached and the motor will not stop.**

---

## **Encoder Error Detection and Protection**

The AX2550 contains an Encoder detection and protection mechanism that will cause the controller to halt if no motion is detected on either Encoder while a power level of 25% or higher is applied to the motor. If such an error occurs, the controller will halt permanently until its power is cycled or it is reset. An Encoder error is one of the conditions that is signaled by the LED rapidly flashing an "8" character (see "Permanent Faults" on page 106).

---

## **Adding Safety Limit Switches**

The Position mode depends on the position sensor providing accurate position information. If the potentiometer is damaged or one of its wire is cut, the motors may spin continuously in an attempt to reach a fictitious position. In many applications, this may lead to serious mechanical damage.

To limit the risk of such breakage, it is recommended to add limit switches that will cause the motors to stop if unsafe positions have been reached independent of the potentiometer reading.

If the controller is equipped with and Encoder module, the simplest solution is to implement limit switches as shown in "Wiring Optional Limit Switches" on page 78. This wiring can be used whether or not Encoders are used for feedback.

If no Encoder module is present, an alternate method is shown in Figure 59. This circuit uses Normally Closed limit switches in series on each of the motor terminals. As the motor reaches one of the switches, the lever is pressed, cutting the power to the motor. The diode in parallel with the switch allows the current to flow in the reverse position so that the motor may be restarted and moved away from that limit.

The diode polarity depends on the particular wiring and motor orientation used in the application. If the diode is mounted backwards, the motor will not stop once the limit switch lever is pressed. If this is the case, reverse the diode polarity.

The diodes may be eliminated, but then it will not be possible for the controller to move the motor once either of the limit switches has been triggered.

The main benefit of this technique is its total independence on the controller's electronics and its ability to work in practically all circumstances. Its main limitation is that the switch and diode must be capable of handling the current that flows through the motor. Note that the current will flow through the diode only for the short time needed for the motor to move away from the limit switches.

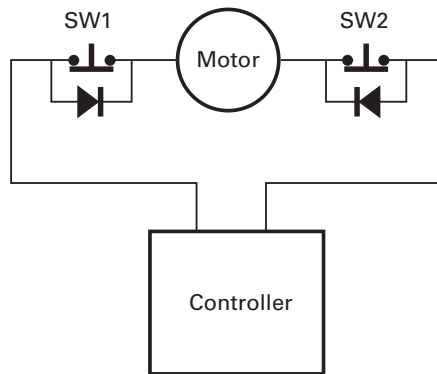


FIGURE 59. Safety limit switches interrupting power to motors

Another method uses the AX2550's Emergency Stop input to shut down the controller if any of the limit switches is tripped. Figure 60 shows the wiring diagram used in this case. Each of the limit switches is a Normally Open switch. Two of these switches are typically required for each motor. Additional switches may be added as needed for the second motor and/or for a manual Emergency Stop. Since very low current flows through the switches, these can be small, low cost switches.

The principal restriction of this technique is that it depends on the controller to be fully functioning, and that once a switch is activated, the controller will remain inactive until the switch is released. In most situations, this will require manual intervention. Another limitation is that both channels will be disabled even if only one channel caused the fault.

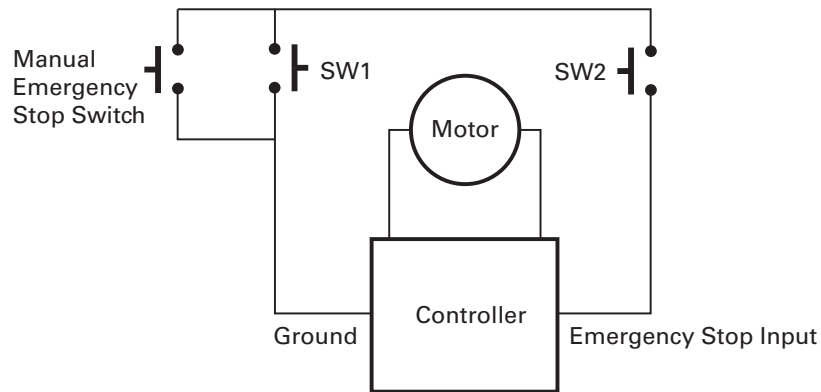


FIGURE 60. Safety limit using AX2550's Emergency Stop input

## **Important Warning**

**Limit switches must be used when operating the controller in Position Mode. This will significantly reduce the risk of mechanical damage and/or injury in case of damage to the position sensor or sensor wiring.**

## **Using Current Limiting as Protection**

It is a good idea to set the controller's current limit to a low value in order to avoid high current draws and consequential damage in case the motor does not stop where expected. Use a value that is no more than 2 times the motor's draw under normal load conditions.

## **Control Loop Description**

The AX2550 performs the Position mode using a full featured Proportional, Integral and Differential (PID) algorithm. This technique has a long history of usage in control systems and works on performing adjustments to the Power Output based on the difference measured between the desired position (set by the user) and the actual position (captured by the position sensor).

Figure 61 shows a representation of the PID algorithm. Every 16 milliseconds, the controller measures the actual motor position and subtracts it from the desired position to compute the position error.

The resulting error value is then multiplied by a user selectable Proportional Gain. The resulting value becomes one of the components used to command the motor. The effect of this part of the algorithm is to apply power to the motor that is proportional with the distance between the current and desired positions: when far apart, high power is applied, with the power being gradually reduced and stopped as the motor moves to the final position. The Proportional feedback is the most important component of the PID in Position mode.

A higher Proportional Gain will cause the algorithm to apply a higher level of power for a given measured error, thus making the motor move quicker. Because of inertia, however, a faster moving motor will have more difficulty stopping when it reaches its desired position. It will therefore overshoot and possibly oscillate around that end position.

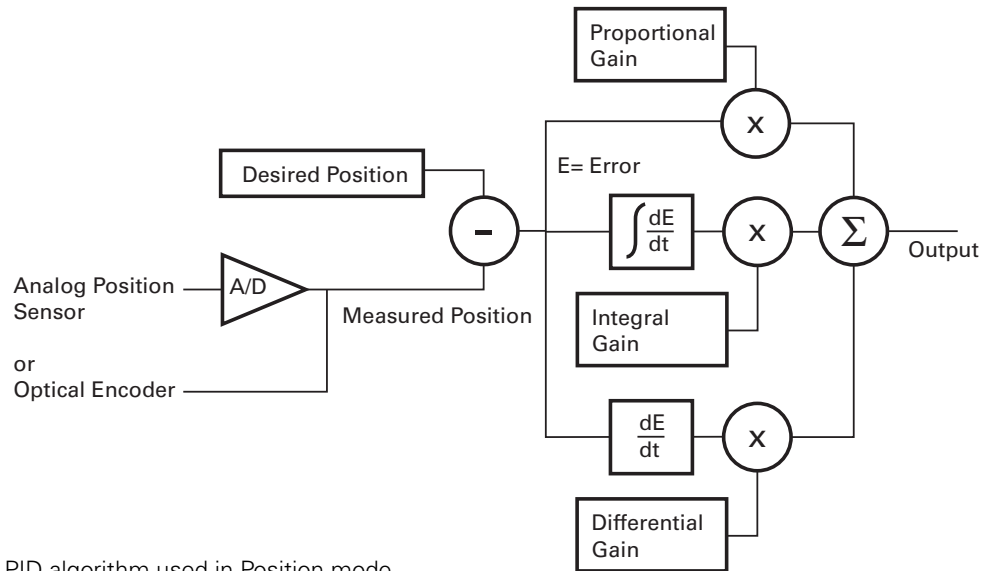


FIGURE 61. PID algorithm used in Position mode

The Differential component of the algorithm computes the changes to the error from one 16 ms time period to the next. This change will be a relatively large number every time an abrupt change occurs on the desired position value or the measured position value. The value of that change is then multiplied by a user-selectable Differential Gain and added to the output. The effect of this part of the algorithm is to give a boost of extra power when starting the motor due to changes to the desired position value. The differential component will also help dampen any overshoot and oscillation.

The Integral component of the algorithm performs a sum of the error over time. In the position mode, this component helps the controller reach and maintain the exact desired position when the error would otherwise be too small to energize the motor using the Proportional component alone. Only a very small amount of Integral Gain is typically required in this mode.

## PID tuning in Position Mode

As discussed above, three parameters - Proportional Gain, Integral Gain and Differential Gain - can be adjusted to tune the position control algorithm. The ultimate goal in a well tuned PID is a motor that reaches the desired position quickly without overshoot or oscillation.

Because many mechanical parameters such as motor power, gear ratio, load and inertia are difficult to model, tuning the PID is essentially a manual process that takes experimentation.

The Roborun PC utility makes this experimentation easy by providing one screen for changing the Proportional, Integral and Differential gains and another screen for running and monitoring the motors.

When tuning the motor, first start with the Integral Gain at zero, increasing the Proportional Gain until the motor overshoots and oscillates. Then add Differential gain until there is no more overshoot. If the overshoot persists, reduce the Proportional Gain. Add a minimal amount of Integral Gain. Further fine tune the PID by varying the gains from these positions.

To set the Proportional Gain, which is the most important parameter, use the Roborun utility to observe the three following values:

- Command Value
- Actual Position
- Applied Power

With the Integral Gain set to 0, the Applied Power should be:

$$\text{Applied Power} = (\text{Command Value} - \text{Actual Position}) * \text{Proportional Gain}$$

Experiment first with the motor electrically or mechanically disconnected and verify that the controller is measuring the correct position and is applying the expected amount of power to the motor depending on the command given.

Verify that when the Command Value equals the Actual Position, the Applied Power equals to zero. Note that the Applied Power value is shown without the sign in the PC utility.

In the case where the load moved by the motor is not fixed, the PID must be tuned with the minimum expected load and tuned again with the maximum expected load. Then try to find values that will work in both conditions. If the disparity between minimal and maximal possible loads is large, it may not be possible to find satisfactory tuning values.

Note that the AX2550 uses one set of Proportional, Integral and Differential Gains for both motors, and therefore assumes that similar motors, mechanical assemblies and loads are present at each channel.





## SECTION 9

# Closed Loop Speed Mode

This section discusses the AX2550 Close Loop Speed mode.

---

## Mode Description

In this mode, an analog or digital speed sensor measures the actual motor speed and compares it to the desired speed. If the speed changes because of changes in load, the controller automatically compensates the power output. This mode is preferred in precision motor control and autonomous robotic applications.

The AX2550 incorporates a full-featured Proportional, Integral, Differential (PID) control algorithm for quick and stable speed control.

---

## Selecting the Speed Mode

The speed mode is selected by changing the Motor Control parameter in the controller to either:

- A and B Closed Loop Speed, Separate
- A and B Closed Loop Speed, Mixed
- A Closed Loop Speed, B Position

Note that in the last selection, only the first motor will operate in the Closed Loop Speed mode.

Changing the parameter to select this mode is done using the Roborun Utility. See “Loading, Changing Controller Parameters” on page 181.

For safety reasons and to prevent this mode from being accidentally selected, Closed Loop Speed modes **CANNOT** be selected by configuring the controller using the built-in switches and display.

---

## Using Optical Encoder for Speed Feedback (AX2850 only)

Digital Optical Encoders may be used to capture accurate motor speed. This capability is only available on controllers fitted with the optional encoder module.

Detailed information on how to install and wire optical encoders is provided at "Installing, Connecting and Using the Encoder Module" on page 71.

If using optical encoders, omit the Analog Tachometer discussion in this section and resume reading from "Control Loop Description" on page 100. Optical Encoders require special handling. See "Installing, Connecting and Using the Encoder Module" on page 71 for a detailed discussion.

---

## Tachometer or Encoder Mounting

Proper mounting of the speed sensor is critical for an effective and accurate speed mode operation. Figure 62 shows a typical motor and tachometer or encoder assembly.

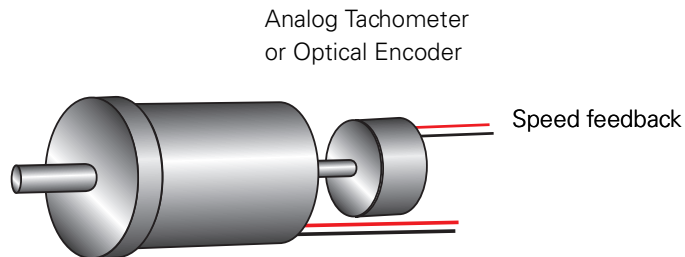


FIGURE 62. Motor and speed sensor assembly needed for Close Loop Speed mode

---

## Tachometer wiring

The tachometer must be wired so that it creates a voltage at the controller's analog input that is proportional to rotation speed: 0V at full reverse, +5V at full forward, and 0 when stopped.

Connecting the tachometer to the controller is as simple as shown in the diagram below.

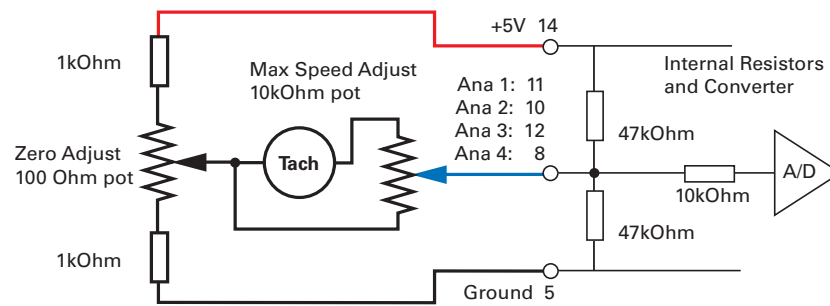


FIGURE 63. Tachometer wiring diagram

## Speed Sensor and Motor Polarity

The tachometer or encoder polarity (i.e. which rotation direction produces a positive or negative speed information) is related to the motor's rotation speed and the direction the motor turns when power is applied to it.

In the Closed Loop Speed mode, the controller compares the actual speed, as measured by the tachometer, to the desired speed. If the motor is not at the desired speed and direction, the controller will apply power to the motor so that it turns faster or slower, until reached.

## Important Warning:

**If there is a polarity mismatch, the motor will turn in the wrong direction and the speed will never be reached. The motor will turn continuously at full speed with no way of stopping it other than cutting the power or hitting the Emergency Stop buttons.**

Determining the right polarity is best done experimentally using the Roborun utility (see "Using the Roborun Configuration Utility" on page 177) and following these steps:

1. Disconnect the controller's Motor Power.
2. Configure the controller in Open Loop Mode using the PC utility. This will cause the motors to run in Open Loop for now.
3. Launch the Roborun utility and click on the Run tab. Click the "Start" button to begin communication with the controller. The tachometer values will be displayed in the appropriate Analog input value boxes which will be labeled Ana 1 and Ana 2. If encoders are used, look for the reported speed value in the Enc boxes.
4. Verify that the motor sliders are in the "0" (Stop) position.
5. If a tachometer is used, verify that the measured speed value read is 0 when the motors are stopped. If not, trim the "0" offset potentiometer.
6. Apply power to the Motor Power wires. The motor will be stopped.

7. Move the cursor of the desired motor to the right so that the motor starts rotating, and verify that a positive speed is reported. Move the cursor to the left and verify that a negative speed is reported.
8. If the tachometer or encoder polarity is the same as the applied command, the wiring is correct.
9. If the tachometer polarity is opposite of the command polarity, then either reverse the motor's wiring, or reverse the tachometer wires. If an encoder is used, swap its CHA and ChB outputs
10. If a tachometer is used, proceed to calibrate the Max Closed Loop speed.
11. Set the controller parameter to the desired Closed Loop Speed mode using the Roborun utility.

---

## Adjust Offset and Max Speed

For proper operation, the controller must see a 0 analog speed value (2.5V voltage on the analog input).

To adjust the 0 value when the motors are stopped, use the Roborun utility to view the analog input value while the tachometer is not turning. Move the 0 offset potentiometer until a stable 0 is read. This should be right around the potentiometer's middle position.

The tachometer must also be calibrated so that it reports a +127 or -127 analog speed value (5V or 0V on the analog input, respectively) when the motors are running at the maximum desired speed in either direction. Since most tachometers will generate more than +/- 2.5V, a 10kOhm potentiometer must be used to scale its output.

To set the potentiometer, use the Roborun utility to run the motors at the desired maximum speed while in Open Loop mode (no speed feedback). While the tachometer is spinning, adjust the potentiometer until the analog speed value read is reaching 126.

Note: The maximum desired speed should be lower than the maximum speed that the motors can spin at maximum power and no load. This will ensure that the controller will be able to eventually reach the desired speed under most load conditions.

## Important Warning:

**It is critically important that the tachometer and its wiring be extremely robust. If the tachometer reports an erroneous voltage or no voltage at all, the controller will consider that the motor has not reached the desired speed value and will gradually increase the applied power to the motor to 100% with no way of stopping it until power is cut off or the Emergency Stop is activated.**

---

## Control Loop Description

The AX2550 performs the Closed Loop Speed mode using a full featured Proportional, Integral and Differential (PID) algorithm. This technique has a long history of usage in control systems and works on performing adjustments to the Power Output based on the differ-

ence measured between the desired speed (set by the user) and the actual position (captured by the tachometer).

Figure 64 shows a representation of the PID algorithm. Every 16 milliseconds, the controller measures the actual motor speed and subtracts it from the desired position to compute the speed error.

The resulting error value is then multiplied by a user selectable Proportional Gain. The resulting value becomes one of the components used to command the motor. The effect of this part of the algorithm is to apply power to the motor that is proportional with the difference between the current and desired speed: when far apart, high power is applied, with the power being gradually reduced as the motor moves to the desired speed.

A higher Proportional Gain will cause the algorithm to apply a higher level of power for a given measured error thus making the motor react more quickly to changes in commands and/or motor load.

The Differential component of the algorithm computes the changes to the error from one 16 ms time period to the next. This change will be a relatively large number every time an abrupt change occurs on the desired speed value or the measured speed value. The value of that change is then multiplied by a user selectable Differential Gain and added to the output. The effect of this part of the algorithm is to give a boost of extra power when starting the motor due to changes to the desired speed value. The differential component will also greatly help dampen any overshoot and oscillation.

The Integral component of the algorithm perform a sum of the error over time. This component helps the controller reach and maintain the exact desired speed when the error is reaching zero (i.e. measured speed is near to, or at the desired value).

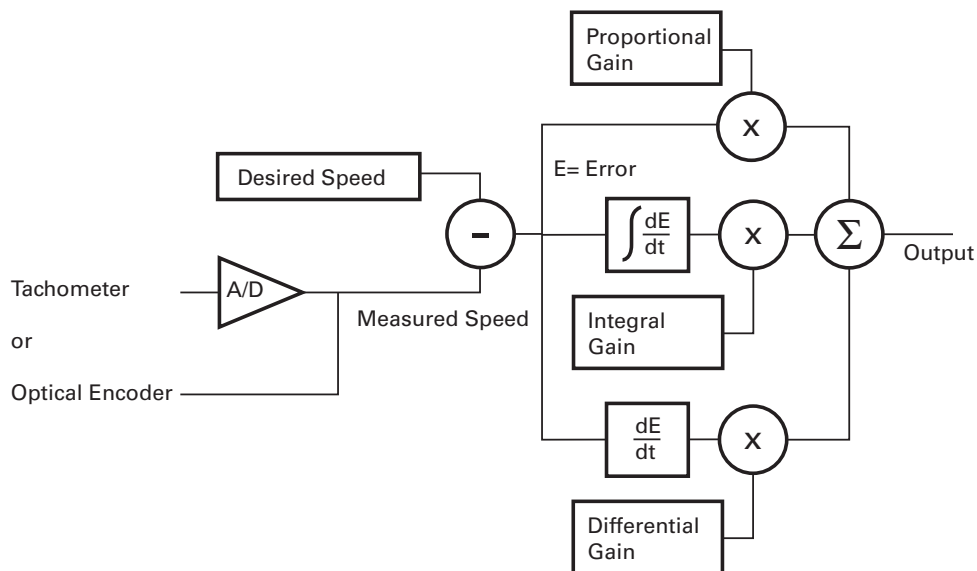


FIGURE 64. PID algorithm used in Speed mode

---

## PID tuning in Speed Mode

As discussed above, three parameters - Proportional Gain, Integral Gain, and Differential Gain - can be adjusted to tune the Closed Loop Speed control algorithm. The ultimate goal in a well tuned PID is a motor that reaches the desired speed quickly without overshoot or oscillation.

Because many mechanical parameters such as motor power, gear ratio, load and inertia are difficult to model, tuning the PID is essentially a manual process that takes experimentation.

The Roborun PC utility makes this experimentation easy by providing one screen for changing the Proportional, Integral and Differential gains and another screen for running and monitoring the motors. First, run the motor with the preset values. Then experiment with different values until a satisfactory behavior is found.

In Speed Mode, the Integral component of the PID is the most important and must be set first. The Proportional and Differential component will help improve the response time and loop stability.

In the case where the load moved by the motor is not fixed, tune the PID with the minimum expected load and tune it again with the maximum expected load. Then try to find values that will work in both conditions. If the disparity between minimal and maximal possible loads is large, it may not be possible to find satisfactory tuning values.

Note that the AX2550 uses one set of Proportional Integral and Differential Gains for both motors and therefore assumes that similar motors, mechanical assemblies and loads are present at each channel.

## SECTION 10

# Normal and Fault Condition LED Messages

This section discusses the meaning of the various messages and codes that may be displayed on the LED display during normal operation and fault conditions.

---

## Use of the LED Display

The AX2550 uses a single 7-segment LED display to report a number of operating or fault conditions. The type of reported information depends on the controller's operating context:

During normal motor operation:

- Motor direction

During Parameter Settings

- Selected parameter and its value

In R/C mode with Radio off

- No control message

During Error condition

- Error condition (overheat, emergency stop, short circuit)

During Self-Test mode

- Software revision number
- Motor direction

### Motor Direction Status

When the controller is running, two pairs of LED segments are directly related to command informations to the Power Output stage. The position and meaning of the segments are shown in the figure below.

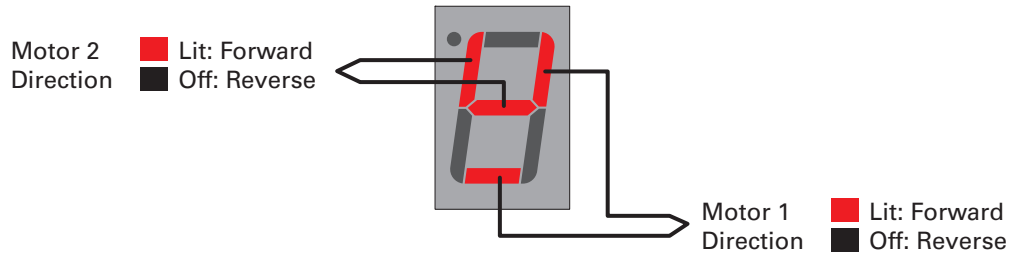


FIGURE 65. Each command bit is wired to 2 LED segments

Note that the display does not provide Power information. Therefore it is possible that the motor be stopped while the display indicates that the direction is forward. In such a situation controller is set to apply the power in the forward direction to the output stage but the motor is stopped because the applied power is zero.

The LED can display a total of 5 patterns summarized in Table 18.

TABLE 18. Motor Commands and resulting display

Possible Display	Motor 1	Motor 2	Comment
	 Forward	 Forward	Is also displayed when controller is active with a 0 command on each channel (i.e. motors at speed 0)
	 Reversed	 Forward	



TABLE 18. Motor Commands and resulting display

Possible Display	Motor 1	Motor 2	Comment
	 Forward	 Reversed	
	 Reversed	 Reversed	
	<b>MOSFET Transistors are OFF</b> Motors are freewheeling		Signals a temporary Off condition in case of Overvoltage, undervoltage or Overtemperature condition (see page 38), or if Dead-man switch is activated (see page 52)
<b>Rapidly Flashing</b> 	<b>MOSFET Transistors are OFF</b> Motors are freewheeling		Controller is permanently Off after an Encoder Error, Short Circuit detection or Emergency Stop activation. Controller must be restarted.

## Fault Messages

The AX2550 uses the LED display to report fault conditions. When these messages are displayed, the motors are normally stopped.

## No Control

This message is displayed in the R/C mode to indicate that no valid radio signal has been detected at its inputs or that radio signal has been lost. When the controller is configured in the mixed mode, signals on channel 1 and channel 2 must be present to enable the controller. When the controller is configured in the separate mode, a signal received on either channel will enable the controller. When enabled, the controller will display the normal motor status described previously.

The “no control” message is displayed using the following sequence of digits.



FIGURE 66. No Radio signal scrolling message

## Temporary Faults

Temporary Faults are condition that cause the controller's Power Output stage to turn Off and remain off for as long as the fault is present. Temporary Faults are indicated with the LED displaying a solid "8". Conditions that cause temporary faults are:

- Overvoltage
- Undervoltage
- Overtemperature condition (see page 38)
- Dead-man activation (see page 52)

## Permanent Faults

Permanent Faults are irrecoverable error condition. When such a condition is detected, the controller's Power Output stage is turned Off for safety reasons and remains Off until it is reset or powered down and up.

Permanent Faults are indicated with a rapidly flashing "8". Condition causing a permanent fault are:

- Encoder Error
- Short Circuit detection
- Emergency Stop activation

Emergency stop condition can be triggered by pressing the Prog and Set button simultaneously, or by activating the Estop/Inv input if enabled.

## Self-Test Display

Self test is a special condition that is entered by holding the Program and Set button immediately after power-on or reset. During self test, the LEDs will display the controller's software revision number by flashing a sequence of digits as shown in the figure below. Then each motor will, in turn, be ramped to maximum forward and maximum reverse. When the motors are operating, the LED will display one of the patterns described in Table 18. The cycle will repeat indefinitely until the controller is powered Off or reset.



FIGURE 67. Example of Software revision number display

---



SECTION 11

# R/C Operation

This section describes the controller's wiring and functions specific to the R/C radio control mode.

## Mode Description

The AX2550 can be directly connected to an R/C receiver. In this mode, the speed or position information is contained in pulses whose width varies proportionally with the joysticks' positions. The AX2550 mode is compatible with all popular brands of R/C transmitters. A third R/C channel can be used to control the On/Off state of two outputs that may be connected to electrical accessories (valves, lights, weapons,...)

The R/C mode provides the simplest method for remotely controlling a robotic vehicle: little else is required other than connecting the controller to the R/C receiver (using the provided cable) and powering it On. For better control and improved safety, the AX2550 can be configured to perform correction on the controls and will continuously monitor the transmission for errors.

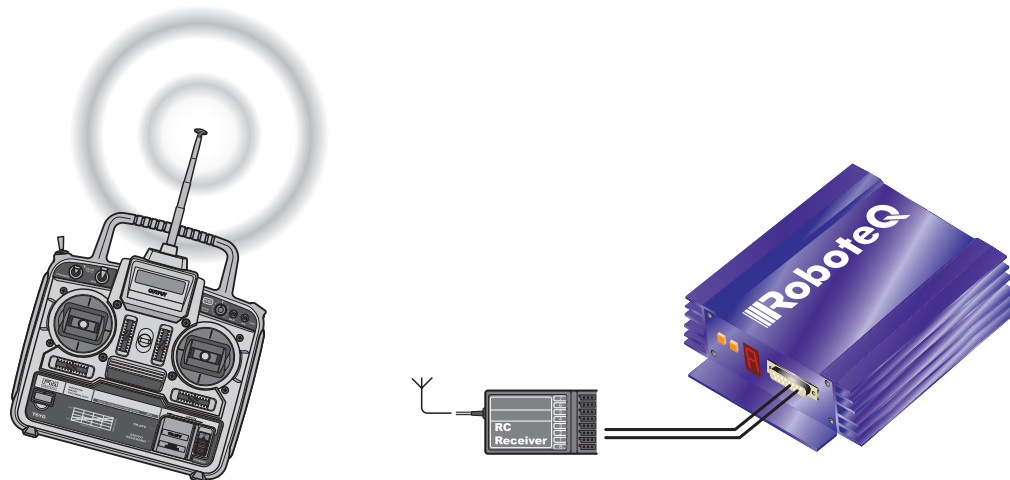


FIGURE 68. R/C radio control mode

## Selecting the R/C Input Mode

The R/C Input Mode is the factory default setting.

If the controller has been previously set to a different Input Mode, it will be necessary to reset it to the R/C mode using one of the following methods:

- Restoring the factory defaults by pressing and holding the **Program** and **Set** buttons while powering on the controller until the LED display flashes
- Setting the “**I**” parameter to the value “**0**” using one of several methods described in the chapters “Configuring the Controller using the Switches” on page 171, “Using the Roborun Configuration Utility” on page 177, and “Accessing & Changing Configuration Parameter in Flash” on page 143.

## Connector I/O Pin Assignment (R/C Mode)

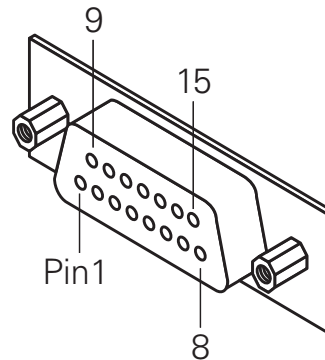


FIGURE 69. Pin locations on the controller’s 15-pin connector

When used in R/C mode, the pins on the controller’s DB15 connector are mapped as described in the table below.

TABLE 19. Connector pin-out in R/C mode

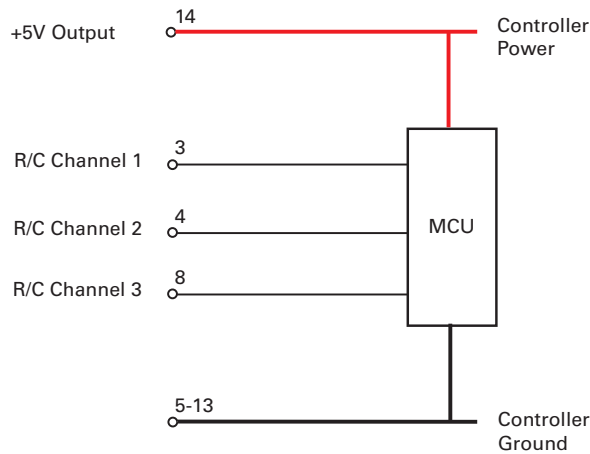
Pin Number	Input or Output	Signal	Description
1 and 9	Output	Output C	2A Accessory Output C
2	Output	RS232 data	RS232 Data Logging Output
3	Input	Ch 1	R/C radio Channel 1 pulses
4	Input	Ch 2	R/C radio Channel 2 pulses
5 and 13	Power Out	Ground	Controller ground (-)
6	GND In	Ground	Optocoupler GND Input Unused in RevB Hardware
7	+5V In	+5V	Optocoupler +5V Input Unused in RevB Hardware

**TABLE 19.** Connector pin-out in R/C mode

Pin Number	Input or Output	Signal	Description
8	Digital In	R/C: Ch 3 / Ana In 4	R/C radio Channel 3 pulses - (Not available when encoder module present)
10	Analog in	Ana in 2	Channel 2 speed or position feedback input
11	Analog in	Ana in 1	Channel 1 speed or position feedback input
12	Analog in	Ana in 3	Unused
14	Power Out	+5V	+5V Power Output (100mA max.)
15	Input	Input EStop/Inv	Emergency Stop or Invert Switch input

## R/C Input Circuit Description

The AX2550 R/C inputs are directly connected to the MCU logic. Figure 70 shows an electrical representation of the R/C input circuit.


**FIGURE 70.** AX2550 R/C Input equivalent circuit

## Supplied Cable Description

The AX2550 is delivered with a custom cable with the following wiring diagram:

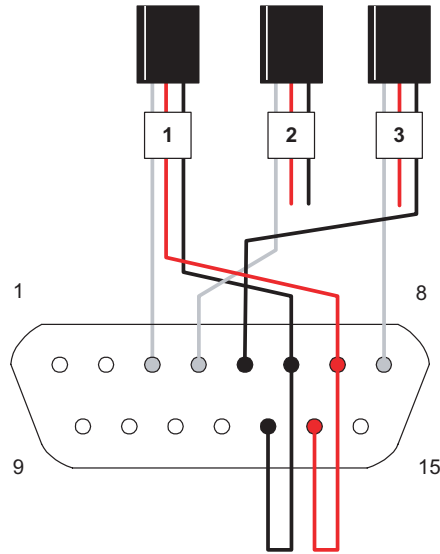


FIGURE 71. RC Cable wiring diagram

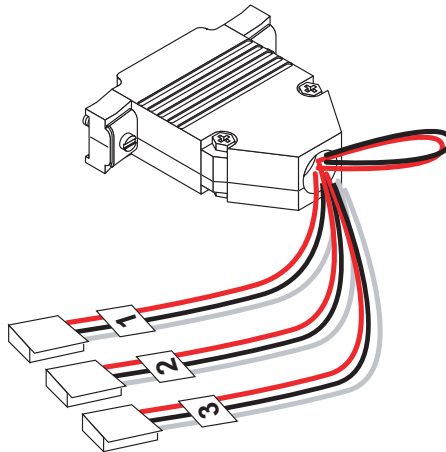


FIGURE 72. RC connection cable

## Powering the Radio from the controller

The 5V power and ground signals that are available on the controller's connector may be used to power the R/C radio. The wire loop is used to bring the controller's power to the the radio as well as for powering the optocoupler stage. Figure 73 below shows the connector wiring necessary to do this. Figure 74 shows the equivalent electrical diagram.



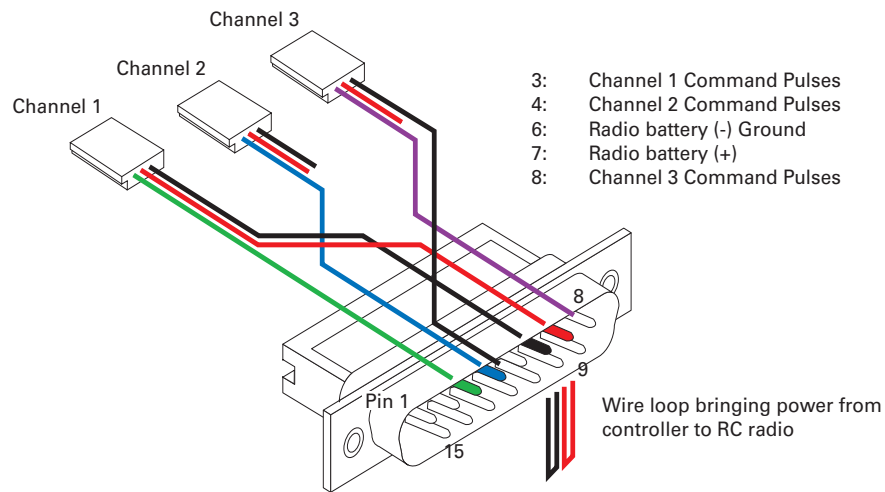


FIGURE 73. Wiring for powering R/C radio from controller

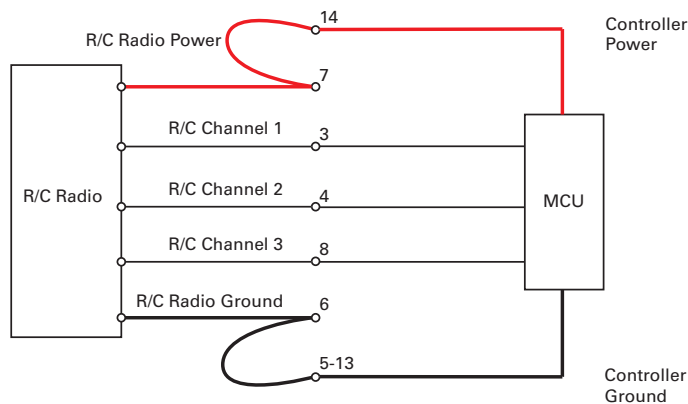


FIGURE 74. R/C Radio powered by controller electrical diagram

## Important Warning

**Do not connect a battery to the radio when in this mode. The battery voltage will flow directly into the controller and cause permanent damage if its voltage is higher than 5.5V.**

This mode of operation is the most convenient and is the one wired in the R/C cable delivered with the controller.

## Connecting to a Separately Powered Radio

This wiring option **must be used** when the controller is used with a RC receiver that is powered by its own separate battery. The red wire in the loop must be cut so that the 5V out from the controller does not flow to the radio, and so that the battery that is connected to the controller does not inject power into the controller. The figure below show the cable with the loop cut. Figure 76 shows the equivalent electrical diagram.

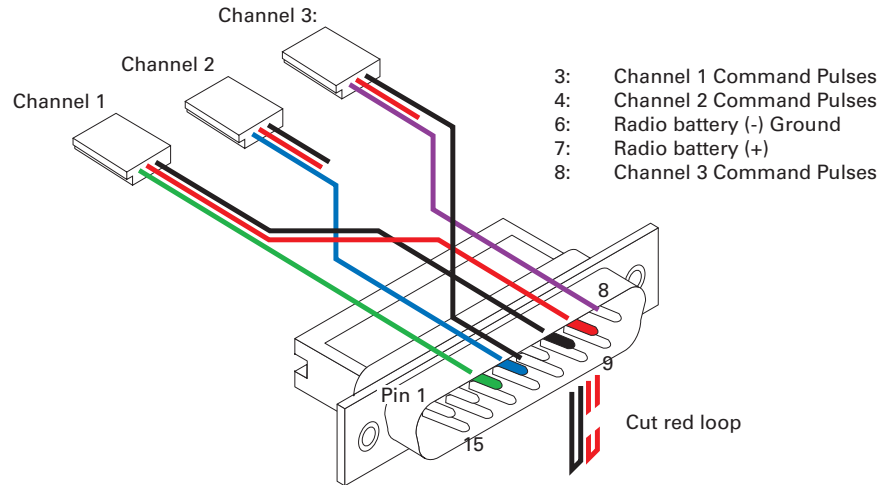


FIGURE 75. Wiring when receiver is powered by its own separate battery

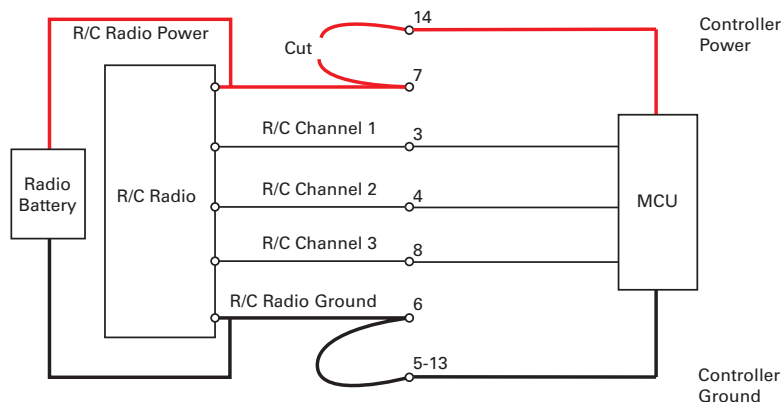


FIGURE 76. Electrical diagram for connection to independently powered RC radio

## Operating the Controller in R/C mode

In this operating mode, the AX2550 will accept commands from a Radio Control receiver used for R/C models remote controls. The speed or position information is communicated to the AX2550 by the width of a pulse from the R/C receiver: a pulse width of 1.0 millisecond.

ond indicates the minimum joystick position and 2.0 milliseconds indicates the maximum joystick position. When the joystick is in the center position, the pulse should be 1.5ms.

Note that the real pulse-length to joystick-position numbers that are generated by your R/C radio may be different than the ideal 1.0ms to 2.0ms discussed above. To make sure that the controller captures the full joystick movement, the AX2550 defaults to the timing values shown in Figure 77. These vales can be changed and stored as new defaults.

For best control accuracy, the AX2550 can be calibrated to capture and use your radio's specific timing characteristics and store them into its internal Flash memory. This is done using a simple calibration procedure described on page 119.

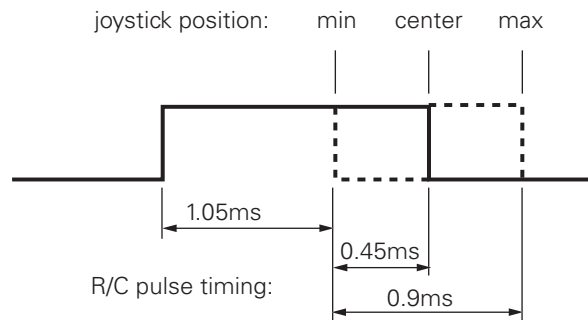


FIGURE 77. Joystick position vs. pulse duration default values

The AX2550 has a very accurate pulse capture input and is capable of detecting changes in joystick position (and therefore pulse width) as small as 0.4%. This resolution is superior to the one usually found in most low cost R/C transmitters. The AX2550 will therefore be able to take advantage of the better precision and better control available from a higher quality R/C radio, although it will work fine with lesser expensive radios as well.

Internally, the measured pulse width is compared to the reference minimum, center and maximum pulse width values. From this is generated a number ranging from -127 (when the joystick is in the min. position), to 0 (when the joystick is in the center position) to +127 (when the joystick is in the max position). This number is then used to set the motors' desired speed or position that the controller will then attempt to reach.

For best results, reliability and safety, the controller will also perform a series of corrections, adjustments and checks to the R/C commands, as described in the following sections.

## Reception Watchdog

Immediately after it is powered on, if in the R/C mode, the controller is ready to receive pulses from the R/C radio and move the motors accordingly.

If no pulses are present, the motors are disabled, and the controller's display will scroll alternatively the letters "no ctrl" as shown in Figure 78 below.



FIGURE 78. "No control" message will scroll when no valid radio signal is present

After powering on the R/C radio receiver and transmitter, and if the wiring is correct, the controller will start receiving pulses. For a preset amount of time, the controller will monitor the pulse train to make sure that they are regular and therefore genuine R/C radio command pulses. After that, the motors are enabled and the LEDs will display a pattern related to the actual motor direction (see Figure , "Motor Direction Status," on page 104).

This power-on Watchdog feature prevents the controller from becoming active from parasite pulses and from moving the motors erratically as a result.

Similarly, if the pulse train is lost while the motors were enabled, the controller will wait a short preset amount of time before it disables the motors. If the pulses reappear during that time, the controller continues without any breaks. If the communication is confirmed to be lost, the "no ctrl" message is displayed again.

Note: the Accessory Outputs C will be turned Off when radio is lost.

## Important Notice about PCM Radios

**PCM radios have their own watchdog circuitry and will output a signal (normally a "safe condition" value) when radio communication is lost. This signal will be interpreted by the AX2550 as a valid command and the controller will remain active. To benefit from the AX2550's radio detection function, you will need to disable the PCM radio watchdog.**

## R/C Transmitter/Receiver Quality Considerations

As discussed earlier in this chapter, the AX2550 will capture the R/C's command pulses with great accuracy. It will therefore be able to take advantage of the more precise joysticks and timings that can be found in higher quality R/C radio, if such added precision is desired in the application.

Another important consideration is the R/C receiver's ability to operate in an electrically noisy environment: the AX2550 switches high current at very high frequencies. Such transients along long battery and motor wires will generate radio frequency noise that may interfere with the R/C radio signal. The effects may include reduced remote control range and/or induced errors in the command pulse resulting in jerky motor operation.

A higher quality PCM R/C transmitter/radio is recommended for all professional applications, as these are more immune to noise and interference.

While a more noise-immune radio system is always desirable, it is also recommended to layout the wiring, the controller, radio and antenna so that as little as possible electrical

noise is generated. Section “Electrical Noise Reduction Techniques” on page 37 provides a few suggestions for reducing the amount of electrical noise generated in your robot.

---

## Joystick Deadband Programming

In order to avoid undesired motor activity while the joysticks are centered, the AX2550 supports a programmable deadband feature. A small deadband is set in the controller by default at the factory. This deadband can be stretched, reduced or eliminated using the Roborun utility or by changing the “**d**” parameter using one of the three methods described in the chapter “Configuring the Controller using the Switches” on page 171. The AX2550 has 8 preset deadband values coded 0 to 7. The value 0 disables the deadband. Other values select a deadband according to the table below. The deadband value applies equally to both joysticks.

The deadband is measured as a percentage of total normal joystick travel. For example, a 16% deadband means that the first 16% of joystick motion in either direction will have no effect on the motors. If the joystick is recalibrated to operate using a shorter travel (see “Joystick Calibration” on page 118), the percent value in the table will not be as accurate.

TABLE 20. Selectable deadband values

<b>Deadband Parameter Value</b>	<b>Deadband as Percent of full Joystick Travel</b>
<b>d = 0</b>	No deadband
<b>d = 1</b>	8%
<b>d = 2</b>	16% - default value
<b>d = 3</b>	24%
<b>d = 4</b>	32%
<b>d = 5</b>	40%
<b>d = 6</b>	46%
<b>d = 7</b>	54%

Note that the deadband only affects the start position at which the joystick begins to take effect. The motor will still reach 100% when the joystick is at its full position. An exaggerated illustration of the effect of the deadband on the joystick action is shown in the Figure 79 below.

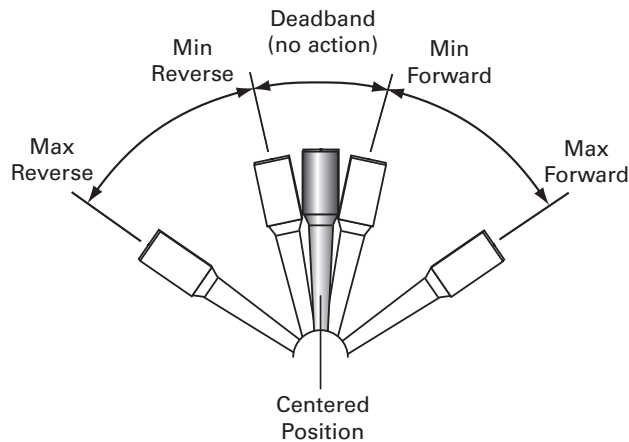


FIGURE 79. Effect of deadband on joystick position vs. motor speed

## Command Control Curves

The AX2550 can also be set to translate the joystick motor commands so that the motors respond differently depending on whether the joystick is near the center or near the extremes. Five different exponential or logarithmic translation curves may be applied. Since this feature applies to the R/C, Analog and RS232 modes, it is described in detail in "Command Control Curves" on page 48, in the General Operation section of this manual.

## Left/Right Tuning Adjustment

When operating in mixed mode with one motor on each side of the robot, it may happen that one motor is spinning faster than the other one at identically applied power, causing the vehicle to pull to the left or to the right.

To compensate for this, the AX2550 can be made to give one side up to 10% more power than the other at the same settings. This capability is described in detail in "Left / Right Tuning Adjustment" on page 49, in the General Operation section of this manual.

## Joystick Calibration

This feature allows you to program the precise minimum, maximum and center joystick positions of your R/C transmitter into the controller's memory. This feature will allow you to use the full travel of your joystick (i.e. minimum = 100% reverse, maximum = 100% forward). It also ensures that the joystick's center position does indeed correspond to a "0" motor command value.

Joystick calibration is also useful for modifying the active joystick travel area. For example, the figure below shows a transmitter whose joystick's center position has been moved back so that the operator has a finer control of the speed in the forward direction than in the reverse position.

There are two methods for calibrating the joysticks:

- The automatic method is the simplest and is based on the controller “learning” the joystick’s parameters using the simple training sequence described below.
- The manual method lets you enter actual timing numbers directly in the controllers flash memory using your PC running the Roborun configuration utility. This method is described in “Loading, Changing Controller Parameters” on page 181.

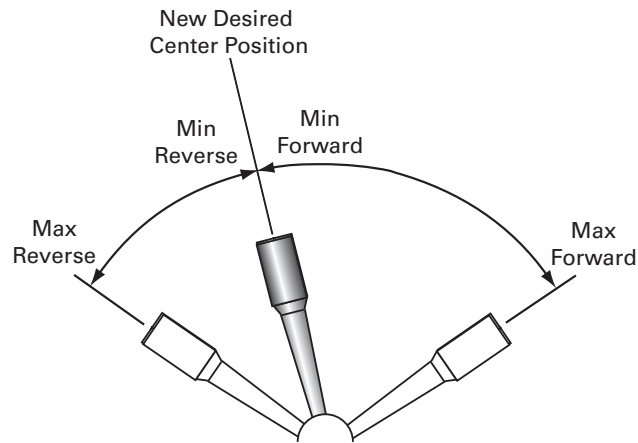


FIGURE 80. Calibration example where more travel is dedicated to forward motion

## Automatic Joystick Calibration

To calibrate the joystick(s) follow these steps:

- Press and hold the **Program** button while resetting or powering up the controller
- After 10 seconds, the controller will enter the Program mode and flash the first parameter (**I**) and its value (**0** if set R/C mode)
- Turn the R/C transmitter and receiver On
- Press the **Program** button several times until the letter “**J**” is displayed. If you miss it, keep pressing the **Program** button for another cycle.
- When the display flashes the letter “**J**” followed by the “-” sign, press the **Set** button
- When the display flashes the letter “**J**” followed by “**o**”, the controller has entered the joystick calibration mode
- Move each joystick to the desired minimum and maximum position several times
- Move the joystick(s) back to the desired center position
- With the joystick(s) in the center position, press the **Program** button to record the change permanently in the controller’s Flash memory.
- Restart the controller by pressing the **Reset** button or cycling the power.

To reset the controller to factory default or to program the joystick’s positions using numerical values, see “Configuring the Controller using the Switches” on page 171 and “Using the Roborun Configuration Utility” on page 177.

Notes:

If you attempt to calibrate the joysticks while the radio is off or not connected to the controller, the calibration data will not change and the previously stored information will continue to be used.

If calibration is performed with only one R/C channel connected to the controller, then only the joystick that is active will be calibrated. The other channel will keep its original settings.

A minimum amount of travel is required between the min, max and center joystick positions. If, while calibrating, the joystick has not been moved far enough from either side of the center position, the controller will automatically include a minimum of travel to ensure proper and safe operation. In most cases, this creates no undesired effect to the driving characteristics of the robot.

R/C calibration only applies to the channel 1 and channel 2 inputs. The accessory activation channel (channel 3) is preset at the factory and cannot be changed.

## **Important Notice**

**To ensure that only stable pulses are present, the R/C transmitter and radio must be On before entering joystick calibration.**

---

## **Activating the Accessory Outputs**

The AX2550 has a general purpose output that may be turned on and off using a third R/C channel on the radio.

Output C is a buffered output capable of driving a 2A device at up to 24V. Details on how to wire this output to user accessories can be found at "Connecting Sensors and Actuators to Input/Outputs" on page 55.

The output is activated by pushing the joystick to the maximum position. The output turns back off when the joystick is returned to the center position.

Note: Channel 3 is not available on the controllers equipped with encoder inputs.



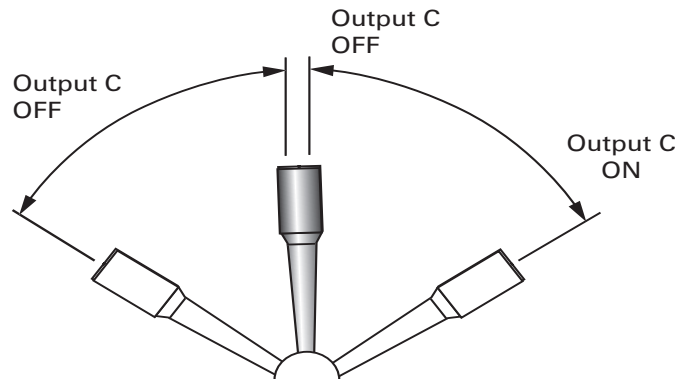


FIGURE 81. Using Channel 3 to activate accessory outputs

## Data Logging in R/C Mode

While in R/C Mode, the AX2550 will continuously send a string of characters on the RS232 output line. This string will contain 12 two-digit hexadecimal numbers representing the following operating parameters.

- Captured R/C Command 1 and 2
- Power Applied to Controller's output stage
- Values applied to Analog inputs 1 and 2
- Amps on channel 1 and 2
- Internal Heat Sink temperatures 1 and 2
- Main Battery voltage
- Internal 12V voltage
- Encoder Speed or Position feedback, if encoder module is present.

The entire string is repeated every 200 milliseconds with the latest internal parameter values. This information can be logged using the Roborun Utility (see "Viewing and Logging Data in Analog and R/C Modes" on page 192). It may also be stored in a PDA that can be placed in the mobile robot.

The string and data format is described in "Analog and R/C Modes Data Logging String Format" on page 166. The serial port's output can be safely ignored if it is not required in the application.

To read the output string while operating the controller with the R/C radio, you must modify the R/C cable to add an RS232 output wire and connector that will be connected to the PC's communication port. Figure 82 and below shows the wiring diagram of the modified R/C cable for connection to a PC.

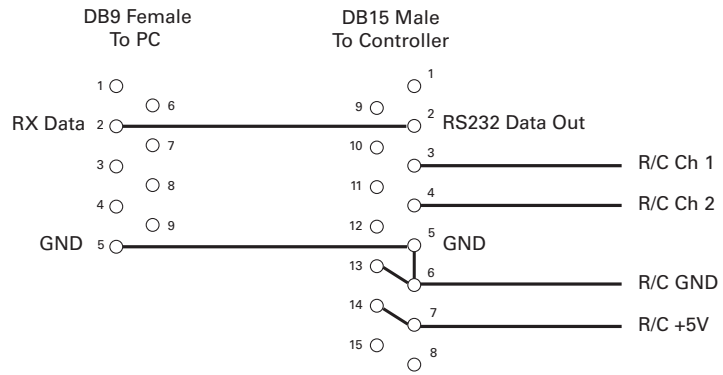


FIGURE 82. Modified R/C cable with RS232 output for data logging to a PC

## SECTION 12

# Analog Control and Operation

This section describes how the motors may be operated using analog voltage commands.

---

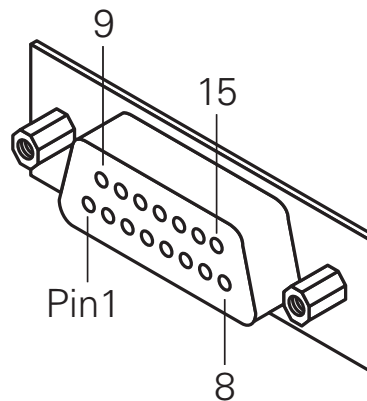
## Mode Description

The AX2550 can be configured to use a 0 to 5V analog voltage, typically produced using a potentiometer, to control each of its two motor channels. The voltage is converted into a digital value of -127 at 0V, 0 at 2.5V and +127 at 5V. This value, in turn, becomes the command input used by the controller. This command input is subject to deadband threshold and exponentiation adjustment. Analog commands can be used to control motors separately (one analog input command for each motor) or in mixed mode.

## Important Notice

**The analog mode can only be used in the Closed Loop speed or position modes when Optical Encoders are used for feedback. Position potentiometers or tachometers cannot be used since there is only one analog input per channel and since this input will be connected to the command potentiometer.**

## Connector I/O Pin Assignment (Analog Mode)



When used in the Analog mode, the pins on the controller's DB15 connector are mapped as described in the table below

TABLE 21. DB15 Connector pin assignment in Analog mode

Pin Number	Signal	Input or Output	Description
1	Output C	Output	2A Accessory Output C (same as pin 9)
2	Data Out	Output	RS232 data output to the PC for data logging
3	Data In	Input	unused
4	Input F	Input	See "Special Use of Accessory Digital Inputs" on page 52
5	Ground Out	Power Output	Controller ground (-)
6	Unused		Unused
7	Unused		Unused
8	Input E / Ana In 4	Input	Channel 2 position feedback input (servo mode)
9	Output C	Output	2A Accessory Output C (same as pin 1)
10	Channel 2 In	Analog in	Channel 2 Command Input
11	Channel 1 In	Analog in	Channel 1 Command Input
12	Analog Input 3	Input	Channel 1 position feedback input (servo mode)
13	Ground Out	Power	Controller ground (-)
14	+5V Out	Power Output	+5V Power Output (100mA max.)
15	Switch Input	Input	Emergency Stop or Invert Switch input

## Connecting to a Voltage Source

The analog inputs expect a DC voltage of 0 to 5V which can be sourced by any custom circuitry (potentiometer, Digital to Analog converter).

The controller considers 2.5V to be the zero position (Motor Off). 0V is the maximum reverse command and +5V is the maximum forward command.

The inputs' equivalent circuit is show in Figure 83 below.

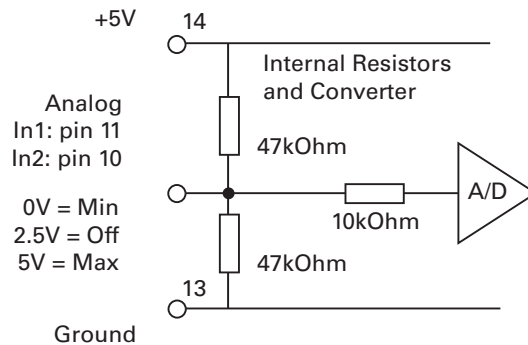


FIGURE 83. Analog input circuit

Notice the two 47K resistors, which are designed to automatically bring the input to a mid-point (Off) position in case the input is not connected. The applied voltage must have sufficient current (low impedance) so that it is not affected by these resistors.

## Connecting a Potentiometer

Figure 84 shows how to wire a potentiometer to the AX2550. By connecting one end to ground and the other to 5V, the potentiometer acts as an adjustable voltage divider. The voltage will thus vary from 0V when the tap is at the minimum position and to 5V when the tap is at the maximum position.

The controller considers 2.5V to be the zero position (Motor Off). 2.5V is the potentiometer's mid point position.

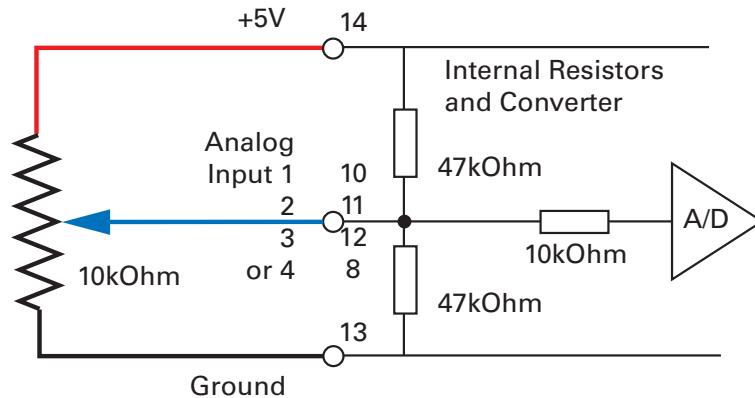


FIGURE 84. Potentiometer connection wiring diagram

The controller includes two 47K ohm resistors pulling the input to a mid-voltage point of 2.5V. When configured in the Analog Input mode, this will cause the motors to be at the Off state if the controller is powered with nothing connected to its analog inputs.

## Important Notice

**The controller will not activate and will display the “no ctrl” message after power up or reset until the analog inputs are at 2.5V**



FIGURE 85. The “no control” message indicates that joystick is not centered at power up

## Selecting the Potentiometer Value

The potentiometer can be of almost any value. Undesirable effects occur, however, if the value is too low or too high.

If the value is low, an unnecessarily high and potentially damaging current will flow through the potentiometer. The amount of current is computed as the voltage divided by the potentiometer's resistance at its two extremes. For a 1K potentiometer, the current is:

$$I = U/R = 5V / 1000 \text{ Ohms} = 0.005A = 5mA$$

For all practical purposes, a 1K potentiometer is a good minimal value.

If the value of the potentiometer is high, then the two 47K resistors built into the controller will distort the reading. The effect is minimal on a 10K potentiometer but is significant on a

100K or higher potentiometer. Figure 86 shows how the output voltage varies at the various potentiometer positions for three typical potentiometer values. Note that the effect is an exponentiation that will cause the motors to start moving slowly and accelerate faster as the potentiometer reaches either end.

This curve is actually preferable for most applications. It can be corrected or amplified by changing the controller's exponentiation parameters (see "Command Control Curves" on page 48).

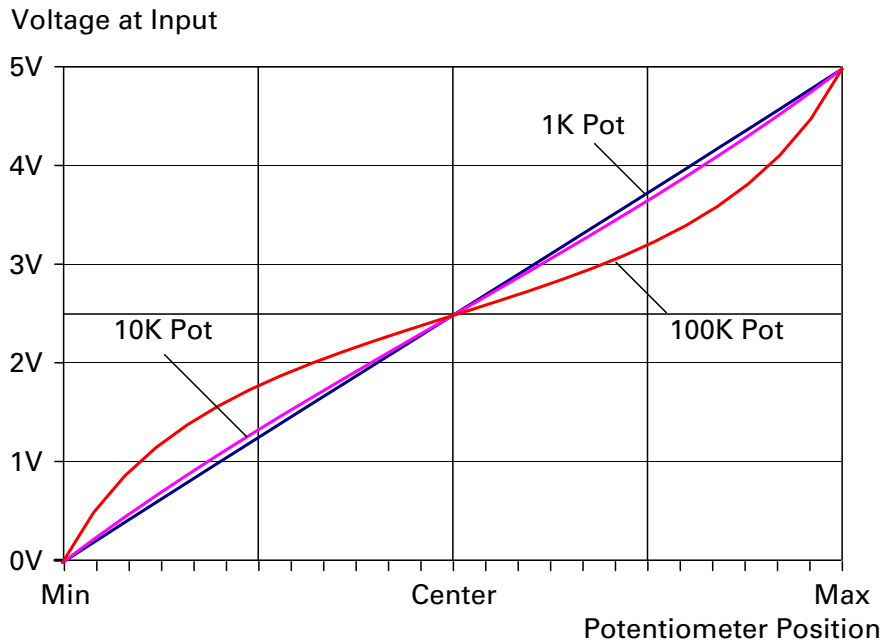


FIGURE 86. Effect of the controller's internal resistors on various potentiometers

## Analog Deadband Adjustment

The controller may be configured so that some amount of potentiometer or joystick travel off its center position is required before the motors activate. The deadband parameter can be one of 8 values, ranging from 0 to 7, which translate into a deadband of 0% to 16%. Even though the deadband will cause some of the potentiometer movement around the center position to be ignored, the controller will scale the remaining potentiometer movement to command the motors from 0 to 100%.

Note that the scaling will also cause the motors to reach 100% at slightly less than 100% of the potentiometer's position. This is to ensure that 100% motor speed is achieved in all circumstances. Table 22 below shows the effect of the different deadband parameter values. Changing the deadband parameter can be done using the controller's switches (see "Configuring the Controller using the Switches" on page 171) or the Roborun utility on a PC (see "Loading, Changing Controller Parameters" on page 181).

TABLE 22. Analog deadband parameters and their effects

Parameter Value	Pot. Position resulting in Motor Power at 0%		Pot. Position resulting in Motor Power at +/-100%	
	0%	2.5V	94%	0.15V and 4.85V
0	0%	2.5V	94%	0.15V and 4.85V
1	0% to 2.4%	2.44V to 2.56V	96%	0.10V and 4.90V
2	0% to 4.7%	2.38V to 2.62V	93%	0.18V and 4.83V
3 (default)	0% to 7.1%	2.32V to 2.68V	95%	0.13V to 4.88V
4	0% to 9.4%	2.27V to 2.74	93%	0.18V and 4.83V
5	0% to 11.8%	2.21V to 2.80V	95%	0.13V to 4.88V
6	0% to 14.2%	2.15V to 2.86V	94%	0.15V and 4.85V
7	0% to 16.5%	2.09V to 2.91V	96%	0.10V and 4.90V

## Important Notice

**Some analog joysticks do not cause the potentiometer to reach either extreme. This may cause the analog voltage range to be above 0V and below 5V when the stick is moved to the extreme, and therefore the controller will not be able to deliver full forward or reverse power.**

## Power-On Safety

When powering on the controller, power will not be applied to the motors until both the Channel 1 and Channel 2 potentiometers have been centered to their middle position (2.5V on each input). This is to prevent the robot or vehicle from moving, in case the joystick was in an active position at the moment the controller was turned on. The “no ctrl” message will scroll on the LED display while the controller is disabled.

## Under Voltage Safety

If the controller is powered through the Power Control input and the motor battery voltage drops below 5V, the controller will be disabled until the analog commands are centered to the midpoint (2.5V on each input).

## Data Logging in Analog Mode

While in Analog Mode, the AX2550 will continuously send a string of characters on the RS232 output line. This string will contain 12 or 13 two-digits hexadecimal number representing the following operating parameters.

- Captured Analog Command 1 and 2
- Power Applied to Controller’s output stage
- Raw analog command values
- Amps on channel 1 and 2



- Internal Heat Sink temperatures 1 and 2
- Main Battery voltage
- Internal 12V voltage
- Encoder Speed or Position (AX2850)

The entire string is repeated every 213 milliseconds with the latest internal parameter values. This information can be logged using the Roborun Utility (see “Viewing and Logging Data in Analog and R/C Modes” on page 192). It may also be stored in a PDA that can be placed in the mobile robot.

The string and data format is described in “Analog and R/C Modes Data Logging String Format” on page 166. The serial port’s output can be safely ignored if it is not required in the application.

To read the output string while operating the controller with an analog command, the cable must be modified to add an RS232 output wire and connector that will be connected to the PC’s communication port. Figure 87 below shows the wiring diagram of the modified cable for connection to a PC or to a PDA, respectively.

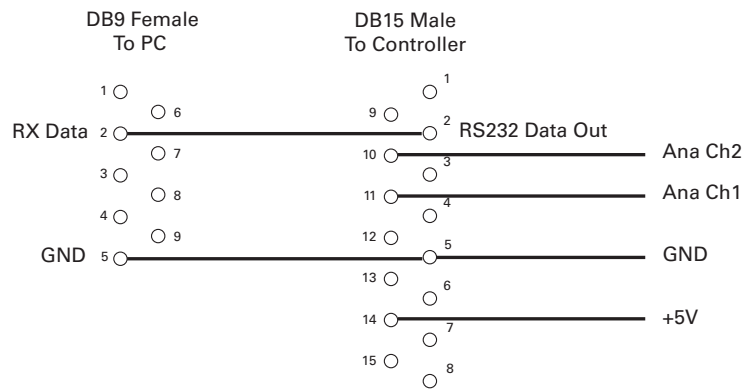


FIGURE 87. Modified Analog cable with RS232 output data logging for PC

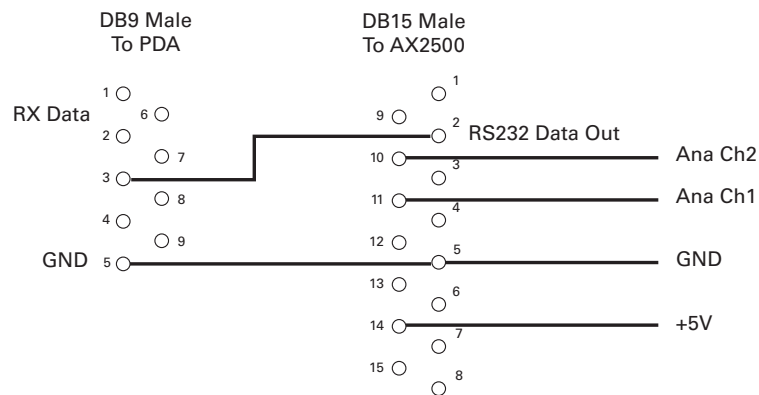


FIGURE 88. Modified Analog cable with RS232 output data logging for PDA



## SECTION 13

# Serial (RS-232) Controls and Operation

This section describes the communication settings and the commands accepted by the AX2550 in the RS232 mode of operations. This information is useful if you plan to write your own controlling software on a PC or microcomputer. These commands will also allow you to send commands manually using a terminal emulation program. If you wish to use your PC simply to set parameters and/or to exercise the controller, you should use the Roborun utility described on page 131.

---

## Use and benefits of RS232

The serial port allows the AX2550 to be connected to microcomputers or wireless modems. This connection can be used to both send commands and read various status information in real-time from the controller. The serial mode enables the design of autonomous robots or more sophisticated remote controlled robots than is possible using the R/C mode. RS232 commands are very precise and securely acknowledged by the controller. They are also the method by which the controller's features can be accessed and operated to their fullest extent.

When connecting the controller to a PC, the serial mode makes it easy to perform simple diagnostics and tests, including:

- Sending precise commands to the motors
- Reading the current consumption values and other parameters
- Obtaining the controller's software revision and date
- Reading inputs and activating outputs
- Setting the programmable parameters with a user-friendly graphical interface
- Updating the controller's software

## Connector I/O Pin Assignment (RS232 Mode)

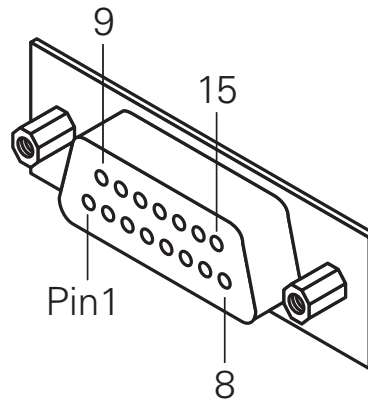


FIGURE 1. Pin locations on the controller's 15-pin connector

When used in the RS232 mode, the pins on the controller's DB15 connector are mapped as described in the table below

**TABLE 23.** DB15 Connector pin assignment in RS232 mode

Pin Number	Input or Output	Signal	Description
1 and 9	Output	Output C	2A Accessory Output C
2	Output	Data Out	RS232 Data from Controller to PC
3	Input	Data In	RS232 Data In from PC
4	Input	Input F	Digital Input F readable RS232 mode Dead man switch activation
5 and 13	Power Out	Ground	Controller ground (-)
6	GND In Unused in RevB Hardware	Ground	Optocoupler GND Input, Connect to pin 5 Unused in RevB Hardware
7	+5V In Unused in RevB Hardware	Power	Optocoupler +5V Input. Connect to pin 14 Unused in RevB Hardware
8	Digital In and Analog In	Input E / Ana in 4	Accessory input E Dead man Switch Input Activate Output C Analog Input 4 in RevB Hardware
10	Analog in	Ana in 2	Channel 2 speed or position feedback input
11	Analog in	Ana in 1	Channel 1 speed or position feedback input
12		Ana in 3	Analog input 3
14	Power Out	+5V	+5V Power Output (100mA max.)
15	Input	Input EStop/Inv	Emergency Stop or Invert Switch input

## Cable configuration

The RS232 connection requires the special cabling as described in the figure below. The 9-pin female connector plugs into the PC (or other microcontroller). The 15-pin male connector plugs into the AX2550.

It is critical that you do not confuse the connector's pin numbering. The pin numbers on the drawing are based on viewing the connectors from the front (facing the sockets or pins). Most connectors have pin numbers molded on the plastic.

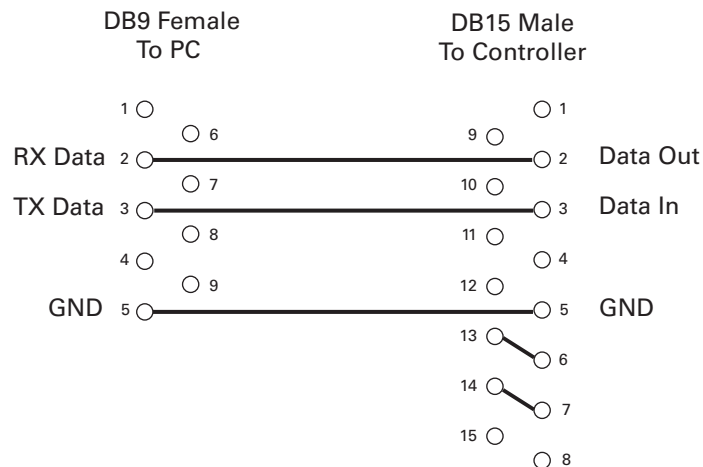


FIGURE 89. PC to AX2550 RS232 cable/connector wiring diagram

## Extending the RS232 Cable

The AX2550 is delivered with a 4-foot cable adapter which may be too short, particularly if you wish to run and monitor the controller inside a moving robot.

RS232 extension cables are available at most computer stores. However, you can easily build one using a 9-pin DB9 male connector, a 9-pin DB9 female connector and any 3-wire cable. These components are available at any electronics distributor. A CAT5 network cable is recommended, and cable length may be up to 100' (30m). Figure 90 shows the wiring diagram of the extension cable.

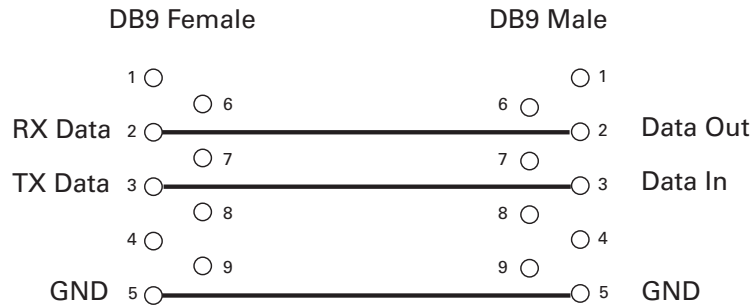


FIGURE 90. RS232 extension cable/connector wiring diagram

## Communication Settings

The AX2550 serial communication port is set as follows:

### 9600 bits/s, 7-bit data, 1 Start bit, 1 Stop bit, Even Parity

Communication is done without flow control, meaning that the controller is always ready to receive data and can send data at any time.

**These settings cannot be changed.** You must therefore adapt the communication setting in your PC or microcomputer to match those of the controller.

## Establishing Manual Communication with a PC

The controller can easily be connected to a PC in order to manually exercise its capabilities. Simply connect the supplied cable to the AX2550 on one end (DB-15 connector) and to a free COM port on the other end (DB-9 connector).

Once connected, you will need a Terminal Emulation program to display the data received from the controller on the PC's screen and to send characters typed on the keyboard to the controller. All Windows PC's come with the Hyperterm terminal emulation software.

Locate the Hyperterm launch icon in the Start button: Programs > Accessories > Communication folder.

You will need to configure Hyperterm to use the COM port to which you have connected the controller (typically COM1) and to configure the communication settings as described in the section above.

To save time and avoid errors, a hyperterm configuration file is automatically installed in your PC's Start button menu when the Roboteq's Roborun utility is installed (See "Downloading and Installing the Utility" on page 177). The configuration file is set to use the

COM1port. You can easily change this setting to a different port from the program's menus.

Note that starting with version 1.9, the Roborun PC utility also includes a Terminal Emulation Console for communicating with the controller using raw data. See "Using the Console" on page 191.

In all cases, immediately after reset or power up, the controller will output a short identity message followed by a software revision number and software revision date as follows:

**Roboteq v1.9b 06/01/07**

**s**

The letter below the prompt message is a code that provides information on the hardware and can be ignored.

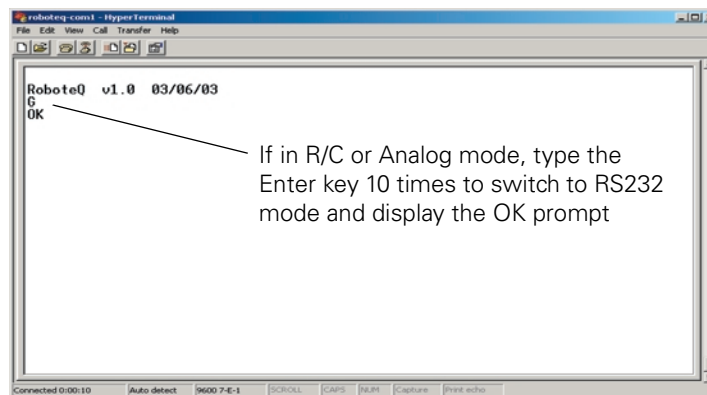


FIGURE 91. Power-on message appearing on Hyperterm

## RS232 Communication with the Encoder Module

The Encoder Module contains its own Microcontroller and firmware in Flash. The Encoder's MCU communicates with the one on the main board of the controller. During normal operations, the two MCUs exchange information as needed, transparently to the user.

During a short time at power up, however, the Encoder's MCU will send data to the main serial port.

The sent data is a separate prompt message which:

- Announces the presence of the encoder MCU
- Outputs its software revision and date
- Outputs a code identifying the module's hardware ID

The serial port settings are described in "Serial (RS-232) Controls and Operation" on page 131.

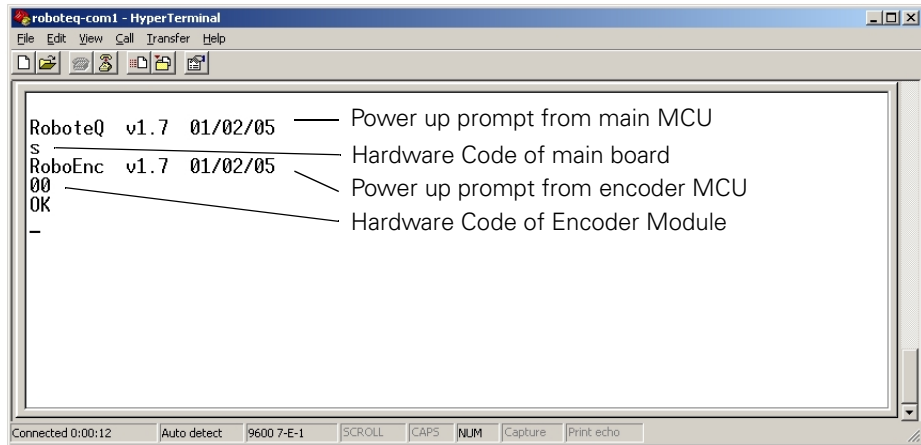


FIGURE 92. Hyperterm session showing power up messages from both MCUs

After this information is sent, the Encoder's MCU will "listen" for approximately 100ms and will enter the In System Programming mode (ISP) if the letter "Z" is sent to it. While in the ISP mode, new software can be loaded into the Encoder's MCU via the controller's main serial port.

Details on software updating are given in section "Updating the Controller's Software" on page 194.

### Entering RS232 from R/C or Analog mode

If the controller is configured in R/C or Analog mode, it will not be able to accept and recognize RS232 commands immediately.

However, the controller will be "listening" to the serial port and will enter the serial mode after it has received 10 continuous "Enter" (Carriage Return) characters. At that point, the controller will output an "OK" message, indicating that it has entered the RS232 mode and that it is now ready to accept commands.

Note that for improved safety, the RS232 watchdog is automatically enabled when entering the RS232 in this way. See "RS-232 Watchdog" on page 138.

When reset again, the controller will revert to the R/C mode or Analog mode, unless the Input Mode parameter has been changed in the meantime.

### Data Logging String in R/C or Analog mode

If the controller is in the R/C or analog mode, immediately after reset it will send a continuous string of characters (one character every 8ms, one entire string every 200ms) containing operating parameters for data logging purposes.

This information can be safely ignored and the controller will still be able to switch to RS232 mode upon receiving 10 continuous Carriage Returns as described above.

The format of the data logging string and its content is described in Figure , "Analog and R/C Modes Data Logging String Format," on page 166



## RS232 Mode if default

If the controller is configured in RS232 mode, it will automatically be in the RS232 mode upon reset or power up.

In this case, the “OK” message is sent automatically, indicating that the controller is ready to accept commands through its serial port.

---

## Commands Acknowledge and Error Messages

The AX2550 will output characters in various situations to report acknowledgements or error conditions as listed below.

### Character Echo

At the most fundamental level, the AX2550 will echo back to the PC or Microcontroller every valid character it has received. If no echo is received, one of the following is occurring:

- the controller is not in the RS232 mode
- the controller is Off
- the controller may be defective

### Command Acknowledgement

The AX2550 will acknowledge commands in one of two ways:

For commands that cause a reply, such as a speed or amps queries, the reply to the query must be considered as the command acknowledgement.

For commands where no reply is expected, such as speed setting, the controller will issue a “plus” character (+) after every command as an acknowledgment.

### Command Error

If a command or query has been received with errors or is wrong, the control will issue a “minus” character (-) to indicate the error.

If the controller issues the “-” character, it should be assumed that the command was lost and that it should be repeated.

### Watchdog time-out

If the RS232 watchdog is enabled, the controller will stop the motors and issue a “W” character if it has not received a valid character from the PC or microcontroller within the past 1 seconds.

## RS-232 Watchdog

For applications demanding the highest operating safety, the controller may be configured to automatically stop the motors (but otherwise remain fully active) if it fails to receive a character on its RS232 port for more than 1 seconds.

The controller will also send a "W" character every second to indicate to the microcomputer that such a time-out condition has occurred.

The character does not need to be a specific command, but any ASCII code, including invisible ones.

The RS232 watchdog is enabled or disabled depending on the value of the "Input Command Mode" parameter.

The RS232 watchdog is automatically enabled when entering the RS232 mode from the RC or from the Analog modes (see "Entering RS232 from R/C or Analog mode" on page 136)

## Controller Commands and Queries

AX2550 commands and queries are composed of a series of 2 or 4 characters followed by the "enter" (carriage return) code.

The controller will send back (echo) every character it is receiving. By checking that the returned character is the same as the one sent, it is possible to verify that there has been no error in communication.

After a command has been received and properly executed, the controller will send the "+" character.

If a command has been received with errors or bad parameters, the controller will send the "-" character.

The table below lists the AX2550 RS232 commands and queries

TABLE 24. Controller's basic Commands and Queries

Command	Type	Description
%rrrrrr	Command	Reset Controller
!Ann	Command	Channel 1, forward command to value nn
!ann	Command	Channel 1, reverse command to value nn
!Bnn	Command	Channel 2, forward command to value nn
!bnn	Command	Channel 2, reverse command to value nn
!C	Command	Turn Accessory Output C n
!c	Command	Turn Accessory Output C Off
?a or ?A	Query	Read Battery Amps
?v or ?V	Query	Read Power Level applied to motors
?p or ?P	Query	Read Analog Inputs 1 and 2
?r or ?R	Query	Read Analog Inputs 3 and 4

TABLE 24. Controller's basic Commands and Queries

<b>Command</b>	<b>Type</b>	<b>Description</b>
?m or ?M	Query	Read Heatsink Temperature
?e or ?E	Query	Read Battery and Internal Voltage
?i or ?I	Query	Read Digital Inputs
?k or ?K	Query	Quick read of Encoder Speed or Position if present

## Set Motor Command Value

Description:

Send a speed or position value from 0 to 127 in the forward or reverse direction for a given channel. In mixed mode, channel 1 value sets the common forward and reverse value for both motors, while channel 2 sets the difference between motor 1 and motor 2 as required for steering. In all other modes, channel 1 commands motor 1 and channel 2 commands motor 2.

Syntax:                **!Mnn**

Where **M=**            **A:** channel 1, forward direction  
                          **a:** channel 1, reverse direction  
                          **B:** channel 2, forward direction  
                          **b:** channel 2, reverse direction

Where **nn=**            Speed or position value in 2 Hexadecimal digits from 00 to 7F

Examples:

**!A00**                channel 1 to 0  
**!B7F**                channel 2, 100% forward  
**!a3F**                channel 1, 50% reverse

Notes:

The hexadecimal number **must always contain two digits**. For example, !a5 will not be recognized and the controller will respond with a "-" to indicate an error. The proper command in this case should be !a05.

## Set Accessory Output

Description:

Turn on or off the digital output line on the 15-pin connector. See "AX2550's Inputs and Outputs" on page 57 for details on how to identify and wire these signals.

Syntax:                **!M**

Where:                **M=**  
                          **c:** output C off  
                          **C:** output C on

Examples:

**!C**                    turn C output off  
**!c**                    turn C output on

## Query Power Applied to Motors

Description:

This query will cause the controller to return the actual amount of power that is being applied to the motors at that time. The number is a hexadecimal number ranging from 0 to +127 (0 to 7F in Hexadecimal). In most cases, this value is directly related to the command value, except in the conditions described in the notes below.

Syntax:           **?v or ?V**

Reply:            **nn**  
                  **mm**

Where:           **nn** = motor 1 applied power value  
                  **mm** = motor 2 applied power value

Notes:

The applied power value that is read back from the controller can be different than the command values for any of the following reasons: current limitation is active, motors operate at reduced speed after overheat detection, or mixed mode is currently active.

No forward or reverse direction information is returned by this query. This query is most useful for providing feedback to a microcontroller commanding the controller.

## Query Amps from Battery to each Motor Channel

Description:

This query will cause the controller to return the actual number of Amps flowing from the battery to power each motor. The number is an unsigned Hexadecimal number ranging from 0 to 256 (0 to FF in Hexadecimal).

Syntax:           **?a or ?A**

Reply:            **nn**  
                  **mm**

Where:           **nn** = motor 1 Amps  
                  **mm** = motor 2 Amps

Notes:

The Amps measurement has an approximately 10% precision. Its main purpose is to provide feedback to the controller's current limitation circuitry.

## Important Notice

The current flowing in the motor can be higher than the battery flowing out of the battery. See "Battery Current vs. Motor Current" on page 45.

## Query Analog Inputs

Description:

This query will cause the controller to return the values of the signals present at its two analog inputs. If the controller is used in close-loop speed mode with analog feedback, the

values represent the actual speed measured by the tachometer. When used in position mode, the values represent the actual motor position measured by a potentiometer. In all other modes, the values represent the measured voltage (0 to 5V) applied to the analog inputs. The values are signed Hexadecimal numbers ranging from -127 to +127. The -127 value represents 0V at the input, the 0 value represents 2.5V, and the +127 value represents +5V.

Analog 1 and 2

Syntax: **?p or ?P**

Analog 3 and 4

Syntax: **?r or ?R**

Reply: **nn**  
**mm**

Where: **nn** = analog input 1 (or 3) value, speed or position  
**mm** = analog input 2 (or 4) value, speed or position

Notes:

The command returns a signed hexadecimal number where 0 to +127 is represented by 00 to 7F, and -1 to -127 is represented by FF to 80 respectively.

On controllers prior to RevB, querying Analog inputs 3 and 4 will return a meaningless number.

## Query Heatsink Temperatures

Description:

This query will cause the controller to return values based on the temperature measured by internal thermistors located at each heatsink side of the controller. Because NTC thermistors are non-linear devices, the conversion of the read value into a temperature value requires interpolation and a look up table. Figure 39 on page 69 shows this correlation. Sample conversion software code is available from Roboteq upon request. The values are unsigned Hexadecimal numbers ranging from 0 to 255. The lowest read value represents the highest temperature.

Syntax: **?m or ?M**

Reply: **nn**  
**mm**

Where: **nn** = thermistor 1 read value  
**mm** = thermistor 2 read value

Notes:

The hexadecimal format is intended to be deciphered by a microcontroller. When exercising the controller manually, you may use the Decimal to Hexadecimal conversion table on page 166.

## Query Battery Voltages

Description:

This query will cause the controller to return values based on two internally measured voltages: the first is the Main Battery voltage present at the thick red and black wires. The second is the internal 12V supply needed for the controller's microcomputer and MOSFET

drivers. The values are unsigned Hexadecimal numbers ranging from 0 to 255. To convert these numbers into a voltage figure, use the formulas described in “Internal Voltage Monitoring Sensors” on page 68.

Syntax: **?e or ?E**

Reply: **nn**  
**mm**

Where: **nn** = main battery voltage value  
**mm** = internal 12V voltage value

Notes:

The hexadecimal format is intended to be deciphered by a microcontroller. When exercising the controller manually, refer to the Decimal to Hexadecimal conversion table on page 167.

## Query Digital Inputs

Description:

This query will cause the controller to return the state of the controller’s two accessory inputs (inputs E and F) and the state of the Emergency Stop/Inverted input. See “Connecting Sensors and Actuators to Input/Outputs” on page 55 for information on how to wire and use these signals. The returned values are three sets of two digits with the values 00 (to indicate a 0 or Off state), or 01 (to indicate a 1 or On state).

Syntax: **?i or ?I**

Reply: **nn**  
**mm**  
**oo**

Where: **nn** = Input E status  
**mm** = Input F status  
**oo** = Estop/Invert Switch Input status

Examples:

**?I** Read Input status query  
**01** Controller replies, Input E is On  
**00** Input F is Off  
**01** Emergency stop switch is high (not triggered)

Note: the Input E value is not meaningful on the AX2850 or on the AX2550 with Encoder module and should be discarded.

## Reset Controller

Description:

This command allows the controller to be reset in the same manner as if the reset button were pressed. This command should be used in exceptional conditions only or after changing the controller’s parameters in Flash memory so that they can take effect.

Syntax: **%rrrrrr**

Reply: None. Controller will reset and display prompt message

## Accessing & Changing Configuration Parameter in Flash

It is possible to use RS232 commands to examine and change the controller's parameters stored in Flash. These commands will appear cryptic and difficult to use for manual parameter setting. It is recommended to use the Graphical configuration utility described in "Using the Roborun Configuration Utility" on page 177. Note that many parameters will not take effect until the controller is reset or a special command is sent. The complete list of parameters accessible using these commands is listed in "Automatic Switching from RS232 to RC Mode" on page 165. Reading and writing parameters is done using the following commands:

### Read parameter

Syntax:        **^mm**

Reply:         **DD**

Where         **mm**= parameter number  
               **DD**= current parameter value

Example:  
**^00**            Read value parameter 0  
**01**            Controller replies, value is 01

### Modify parameter

Syntax:        **^mm nn**

Reply:         **+** if command was executed successfully  
               **-** if error

Where         **mm**= parameter number  
               **nn**= new parameter value

Examples:  
**^02 03**        Store 03 into parameter 2

Notes:  
All parameters and values are expressed with 2 hexadecimal digits

No changes will be made and an error will be reported ("-") character) when attempting to read or write a parameter that does not exist or when attempting to store a parameter with an invalid value.

## Apply Parameter Changes

Description:  
Many parameters will take effect only after the controller is reset. This command can be used (instead of resetting the controller) to cause these parameters to take effect after only a ~100ms delay.

Syntax:        **^FF**

Reply:           + Success, changed parameters are now active  
                   - if error

Table 25 below lists the complete set of configuration parameters that may be accessed and changed using RS232 commands.

## Flash Configuration Parameters List

TABLE 25. Configuration parameters in Flash

Location	Description	Active after
^00	Input control mode	Reset
^01	Motor Control mode and Closed Loop Feedback type	Reset or ^FF
^02	Amps limit	Reset or ^FF
^03	Acceleration	Reset or ^FF
^04	Input Switch function	Reset or ^FF
^05	reserved	
^06	Joystick Deadband or Analog Deadband	Reset or ^FF
^07	Exponentiation on channel 1	Instant
^08	Exponentiation on channel 2	Instant
^09	Reserved	
^0A	Left / Right Adjust	Reset or ^FF
^0B	Encoder 1 Time Base when Encoder Present	Reset or ^FF
^0C	Encoder 2 Time Base when Encoder Present	Reset or ^FF
^0D	Reserved	
^0E	Encoder Distance Divider when Encoder Present	Reset or ^FF
^0F	Gain Integral for PID	Reset or ^FF
^10	Gain Diff for PID	Reset or ^FF
^11	Gain Prop for PID	Reset or ^FF
^12	Joystick Center 1 MS	Instant
^13	Joystick Center 1 LS	Instant
^14	Joystick Center 2 MS	Instant
^15	Joystick Center 2 LS	Instant
^16	Joystick Min 1 MS	Instant
^17	Joystick Min 1 LS	Instant
^18	Joystick Min 2 MS	Instant
^19	Joystick Min 2 LS	Instant
^1A	Joystick Max 1 MS	Instant
^1B	Joystick Max 1 LS	Instant
^1C	Joystick Max 2 MS	Instant



TABLE 25. Configuration parameters in Flash

Location	Description	Active after
^1D	Joystick Max 2 LS	Instant
^F0	Amps Calibration Parameter 1	Reset
^F1	Amps Calibration Parameter 2	Reset

These parameters are stored in the controller's Flash memory and are not intended to be changed at runtime.

## Important Notice

The above parameters are stored in the MCU's configuration flash. Their storage is permanent even after the controller is powered off. However, because of the finite number of times flash memories can be reprogrammed (approx. 1000 times), these parameters are not meant to be changed regularly, or on-the-fly.

**All parameters in Flash (except for the Amps calibration) are reset to their default values every time new firmware is loaded into the controller.**

## Input Control Mode

**Address:**            **^00**  
**Access:**            **Read/Write**  
**Effective:**         **After Reset**

This parameter selects the method the controller uses for accepting commands

Value	Mode	See pages
0	R/C Radio mode (default)	page 109
1	RS232, no watchdog	page 109
2	RS232, with watchdog	
3	Analog mode	page 123

## Motor Control Mode

**Address:**            **^01**  
**Access:**            **Read/Write**  
**Effective:**         **After Reset or ^FF**

This parameters selects the various open loop and closed loop operating modes as well as the feedback method.

Bit	Definition	See pages	
2:0	Motor Control Mode	0 = A & B separate speed open loop (default) 1 = A & B mixed speed open loop 2 = A speed open loop, B position 3 = A & B position 4 = A & B separate speed closed loop 5 = A & B mixed speed closed loop 6 = A speed close loop, B position	page 42 page 97 page 85
6:3	Reserved		
6	Ch1 Feedback type	0 = Analog 1 = Encoder	page 97 page 85
7	Ch2 Feedback type	0 = Analog 1 = Encoder	

### Amps Limit

**Address:**         $\wedge 02$   
**Access:**        Read/Write  
**Effective:**     After Reset or  $\wedge FF$

This parameter configures the controller's Amps limit. Note that this limits the amps flowing out of the power supply. Current flowing through the motors may be higher.

Bit	Definition	See pages	
3:0	Coarse Amps	0 = 30A 1 = 45A 2 = 60A 3 = 75A 4 = 90A 5 = 105A (default) 6 = 120A	page 44
7:4	Fine Amps	Subtract this number from the Coarse Amps value	

## Acceleration

**Address:**            ^03  
**Access:**             Read/Write  
**Effective:**          After Reset or ^FF

This parameter configures the rate at which the controller internally changes the command value from the one it was to the one just received.

Bit	Definition	See pages
7:0	0 = very slow 1 = slow (2) = medium-slow (default) 3 = medium 4 = fast 5 = fastest	See "Programmable Acceleration" on page 47 for complete list of acceptable values

## Input Switches Function

**Address:**            ^04  
**Access:**             Read/Write  
**Effective:**          After Reset or ^FF

This parameter enables and configures the effect of the controller's Digital Inputs and other settings.

Bit	Definition	See pages
1:0	Enable and Configure Invert/Estop	(00) = Input Disabled (default) page 51 01 = Input as Emergency Stop page 52 10 = Disabled 11 = Input as Invert Command
2	Output C when Motor On	(0) = No Action (default) page 51 1 = Output C On when either motor is On
3	Encoder Safety	(0) = No Action (default) page 76 1 = Disables Controller if Encoder detects no movement while power is applied to the motors

Bit	Definition	See pages
5:4	Input E	Unavailable when Encoder Module is present (00) = No action (default) 01 = Cut FET power when Input E is Low 10 = Activate output C 11 = Cut FET when Input E is High
7:6	Input F	(00) = No action (default) 01 = Cut FET power when Input E is Low 10 = Activate output C 11 = Cut FET when Input E is High

### RC Joystick or Analog Deadband

**Address:**        ^06  
**Access:**        Read/Write  
**Effective:**     After Reset or ^FF

This parameter configures the amount of joystick or potentiometer motion can take place around the center position without power being applied to the motors.

Bit	Definition	See pages
7:0	Values are for Joystick deadband 0 = no deadband 1 = 8% (2) = 16% (default) 3 = 24% 4 = 32% 5 = 40% 6 = 46% 7 = 54%	page 117 or page 127

### Exponentiation on Channel 1 and Channel 2

**Address:**        ^08 - Channel 1  
                   ^09 - Channel 2  
**Access:**        Read/Write  
**Effective:**     Instantly

This parameter configures the transfer curve that is applied the input command.

<b>Bit</b>	<b>Definition</b>	<b>See pages</b>
7:0	(0) = Linear (no exponentiation - default) 1 = strong exponential 2 = normal exponential 3 = normal logarithmic 4 = strong logarithmic	page 118

### Left/Right Adjust

**Address:**         $\wedge 0B$   
**Access:**        Read/Write  
**Effective:**      After Reset or  $\wedge FF$

This parameter configures the compensation curve when motors are spinning in one direction vs. the other.

<b>Bit</b>	<b>Definition</b>	<b>See pages</b>
7:0	0, 1, ..., 6 = -5.25%, -4.5%, ..., -0.75% (7) = no adjustment (default) 8, ..., D, E** = +0.75, ..., +4.5%, +5.25%	page 49

### Default Encoder Time Base 1 and 2

**Address:**         $\wedge 0B$  - Encoder 1  
                       $\wedge 0C$  - Encoder 2  
**Access:**        Read/Write  
**Effective:**      After Reset or  $\wedge FF$

These parameters are the Encoder Time base values that are loaded after the controller is reset or powered on. Time Bases are used to determine rotation speed depending on the Encoder's resolution. Time Bases can be changed at Runtime using separate commands (see page 152). Time Base values are integer number from 0 to 63. These parameters are only effective on controllers equipped with Encoder Modules.

<b>Bit</b>	<b>Definition</b>	<b>See pages</b>
7:0	0 to 63 (16) default	page 80

## Default Encoder Distance Divider

**Address:**        ^0E  
**Access:**        Read/Write  
**Effective:**      After Reset or ^FF

This parameter is the Encoder's Distance Divider that is loaded after the controller is reset or powered on. The Encoder Distance Divider can be changed at Runtime using separate commands (see page 152). Parameter values are integer number from 0 to 7. This parameter is only effective on controllers equipped with Encoder Modules.

Bit	Definition	See pages
7:0	0 to 63 (16) default	page 80

## Default PID Gains

**Address:**        ^0F - Proportional Gain  
                   ^10 - Integral Gain  
                   ^11 - Derivative Gain  
**Access:**        Read/Write  
**Effective:**      After Reset or ^FF

These parameters are the Gains values that are loaded after the controller is reset or powered on. These Gains apply to both channels. Gains can be changed at Runtime, and values can be different for each channel using separate commands (see page 152).

Gains values are integer number from 0 to 63. This number is divided by 8 internal so that each increment equals 0.125.

Bit	Definition	See pages
7:0	0 to 63 (16) default	page 93 and page 100

## Joystick Min, Max and Center Values

**Address:**        ^12 - Joystick Center 1 MS  
                   ^13 - Joystick Center 1 LS  
                   ^14 - Joystick Center 2 MS  
                   ^15 - Joystick Center 2 LS  
                   ^16 - Joystick Min 1 MS  
                   ^17 - Joystick Min 1 LS  
                   ^18 - Joystick Min 2 MS  
                   ^19 - Joystick Min 2 LS  
                   ^1A - Joystick Max 1 MS  
                   ^1B - Joystick Max 1 LS

**^1C - Joystick Max 2 MS**

**^1D - Joystick Max 2 LS**

**Effective: Instantly**

These parameters are the Gains values that are loaded after the controller is reset or powered on. These Gains apply to both channels. Gains can be changed at Runtime, and values can be different for each channel using separate commands (see page 152).

Gains values are integer number from 0 to 63. This number is divided by 8 internal so that each increment equals 0.125.

Bit	Definition	See pages
7:0	8 bit value. Two registers used to form one 16 bit number for each Joystick parameter.  Default values (in decimal): Min = 4400 Center = 1600 Max = 3200	page 118

## Reading & Changing Operating Parameters at Runtime

It is possible to change several of the controller's operating modes, on-the-fly during normal operation. Unlike the Configuration Parameters that are stored in Flash (see above), the Operating Parameters are stored in RAM and can be changed indefinitely. After reset, the Operating Parameters are loaded with the values stored in the Configuration Parameter flash. They are then changed using RS232 commands.

Use the command following commands to Read/Change the Operating Modes

Syntax: **^mm**            Read Parameters at location mm

**^mm DD**            Write Parameters DD in location DD

mm and DD are Hexadecimal values.

The table below lists the available parameters

TABLE 26. Runtime R/W Parameters list

Location	Function	R/W
^80	Channel 1 Operating Modes	R/W
^81	Channel 2 Operating Modes	R/W
^82	PID Proportional gain 1	R/W
^83	PID Proportional gain 2	R/W
^84	PID Integral gain 1	R/W
^85	PID Integral gain 2	R/W

TABLE 26. Runtime R/W Parameters list

Location	Function	R/W
^86	PID Differential gain 1	R/W
^87	PID Differential gain 2	R/W
^88	PWM frequency	R/W
^89	Controller Status	R Only
^8A	Controller Model	R Only
^8B	Current Amps limit 1	R Only
^8C	Current Amps limit 2	R Only

## Important Notice:

**Do not write in the locations marked as Read Only. Doing so may cause Controller malfunction.**

## Operating Modes Registers

**Address:**        ^80 - Channel 1  
                      ^81 - Channel 2  
**Access:**        Read/Write  
**Effective:**     Instantly

Modifying the bits in the Operating Mode registers will change the controller's operating modes on-the fly. Changes take effect at the controller's next 16ms iteration loop. After reset, these bits get initialized according to the configuration contained in Flash.

Values are in Hexadecimal. Example: 00000101 = Hex 05

TABLE 27. Operating Modes Register Definition

Bit	Function
7 to 3	Not Used
2	0: Open Loop 1: Closed Loop (when in Speed Mode)
1	0: Speed Mode 1: Position Mode
0	0: Analog Feedback 1: Encoder Feedback

## Read/Change PID Values

**Address:**        ^82 - P1  
                      ^83 - I1  
                      ^84 - D1  
                      ^85 - P2



**Access:**        <sup>^86 - I2</sup>  
                   <sup>^87 - D2</sup>  
**Effective:**     **Read/Write**  
                   **Instantly**

The Proportional, Integral and Derivative gain for each channel can be read and changed on-the-fly. This function also provides a mean for setting different PID values for each channel. Actual Gain value is the value contained in the register divided by 8. Changes take effect at the controller's next 16ms iteration loop. After reset, these bits get initialized according to the configuration contained in Flash.

## PWM Frequency Register

**Address:**        <sup>^88</sup>  
**Access:**        **Read/Write**  
**Effective:**     **Instantly**

The controller's default 16kHz PWM Frequency can be changed to a higher value in fine increments. This feature may be used to reduce the interference in case the controller's PWM frequency harmonics are too close to the radio receiver's frequency. The value can be changed at any time and takes effect immediately. The frequency is:

$$15,625 \text{ Hz} * 255 / \text{Register Value}$$

The controller's default frequency provides the best efficiency and should be changed only if absolutely required and only if operating the controller in RS232 or Analog modes. Changes to the PWM frequency will affect the RS232 watchdog timer and PID may need re-tuning.

The controller automatically reverts to the default 16kHz PWM frequency after reset.

## Controller Status Register

**Address:**        <sup>^89</sup>  
**Access:**        **Read Only**  
**Effective:**     **Instantly**

The Controller Status Register can be polled at any time to see if there is a pending fault condition. Any one bit set will cause the controller to turn off the Power Output stage. Conditions marked as Temporary mean that the controller will resume operation as soon as the fault condition disappears. Permanent conditions will cause the controller to remain off until it is reset either by cycling power, pressing the reset button, or sending the %rrrrr command.

TABLE 28. Controller Status Register Definition

Bit	Fault Condition	Effect
0	Overtoltage	Temporary
1	Overtemperature	Temporary
2	Undervoltage	Temporary
3	Manually Forced MOSFETs Off	Temporary
4	Unused	

TABLE 28. Controller Status Register Definition

Bit	Fault Condition	Effect
5	Confirmed Short Circuit	Permanent
6	Confirmed Encoder Error	Permanent
7	Emergency Stop Pressed	Permanent

### Controller Identification Register

**Address:**        ^8A  
**Access:**         Read Only  
**Effective:**       Instantly

This register may be used to query the Controller's model and some of its optional hardware configurations.

TABLE 29. Controller Identification Register Definition

Bit	Model or Function
0	AX500
1	AX1500
2	AX2500
3	AX3500
4	Unused
5	Encoder Present
6	Short Circuit Detection Present
7	Unused

### Current Amps Limit Registers

**Address:**        ^8B - Channel 1  
                   ^8C - Channel 2  
**Access:**         Read Only  
**Effective:**       Instantly

These registers can be polled to view what the Amps limit is at the current time. This limit normally is the one that is preset by the user except when the controller is operating at high temperature, in which case the allowable current drops as temperature rises. See "Temperature-Based Current Limitation" on page 44.

---

## RS232 Encoder Command Set

When present, the Encoder module responds to a dedicated set of commands and queries.

The serial port setting and basic command format is identical to this for all other functions of the controller as described in "Serial (RS-232) Controls and Operation" on page 131.

### Read Encoder Counter

Description:

Read the value of the Encoder counter(s). The number is a signed 32 bit number that may range from -2,147,836,648 to +2,147,836,647. The value is output in Hexadecimal format of value 80000000 to 7FFFFFFF respectively. To speed up communication, only the significant digits are sent in response to a counter query. For example, if the counter contains the value +5 (which is the same number in decimal and hex), the response to the query will be 5 and not 00000005. The formatting algorithm takes into account the number's sign.

Details on the data format are given in section

Counters' values can be read as Absolute or Relative. An Absolute counter read will return the full counter value after every read query. In a Relative counter read, the counter value is immediately cleared immediately after being read so that the next read query returns the new number of counts since the last time the counter was read.

Additionally, in a few of the query modes, the Encoder module returns the sum for both counters. This is useful for measuring the average travelled distance by the right and left wheels of a robotic vehicle.

Syntax:           **?q or Qn**

Where **n=**       **0:** Encoder 1, Absolute  
                  **1:** Encoder 2, Absolute  
                  **2:** Sum of Encoders 1 and 2, Absolute  
  
                  **4:** Encoder 1, Relative  
                  **5:** Encoder 2, Relative  
                  **6:** Sum of Encoders 1 and 2, Relative

Reply:            **nnnnnnnn**

Where:           **nnnnnnnn** = counter value using 1 to 8 Hex digits. See "Counter Read Data Format" on page 163 for format description

Examples:

**?Q0**            Read Encoder 1, Absolute  
**?Q5**            Read Encoder 2, Relative

### Set/Reset Encoder Counters and Destination Registers

Description:

Set one or both counters to zero or a user-defined value. The value is a signed 32 bit number that may range from -2,147,836,648 to +2,147,836,647 (Hexadecimal format of value 80000000 to 7FFFFFFF respectively).

While resetting is a single step command, setting the counters to a non-zero value requires two steps: 1- load a 4 byte buffer (32-bit) with the desired value. 2- Transfer the buffer's content to the counter(s). Loading the buffer can be done using the commands described in "Read / Modify Encoder Module Registers and Parameters" on page 158. The buffer will also be altered after a Counter Read command, in which case it will contain the last read counter value.

Syntax: **!q or !Qn**

Where **n=**

- 0**: Reset Encoder 1 counter
- 1**: Reset Encoder 2 counter
- 2**: Reset both Encoder counters
  
- 4**: Set Encoder 1 counter with value in buffer
- 5**: Set Encoder 2 counter with value in buffer
- 6**: Set both Encoder both counters with value in buffer
  
- 7**: Set Encoder 1 destination register with value in buffer
- 8**: Set Encoder 2 destination register with value in buffer

Reply: **+** if command was properly received and executed  
**-** if an error occurred

Examples:

**!Q2**            Reset both counters  
**!Q5**            Load value contained in buffer into counter 2

**?Q0**, followed by

**!Q1**            Read counter 1 and copy its value into counter 2

## Read Speed

Description:

This query will cause the controller to return the speed computed by the Encoder module. The values are signed Hexadecimal numbers ranging from -127 to +127. The -127 value represents the maximum RPM in the reverse direction. +127 represents the maximum RPM in the forward direction. The relation of this relative number and the actual, absolute RPM value depends on the encoder's resolution and a user programmable Time Base. See "Using the Encoder to Measure Speed" on page 80 for a detailed discussion.

Syntax: **?z or ?Z**

Reply: **nn**  
**mm**

Where: **nn** = speed 1 value  
**mm** = speed 2 value

Notes:

The command returns a signed hexadecimal number where 0 to +127 is represented by 00 to 7F, and -1 to -127 is represented by FF to 80 respectively. The hexadecimal format is intended to be deciphered by a microcontroller. When exercising the controller manually, you may use the Decimal to Hexadecimal conversion table on page 167.

## Read Distance

Description:

This query will cause the controller to return the distance between the current position and the value in the destination register. The values are signed Hexadecimal numbers ranging from -127 to +127. The -127 value represents the relative distance according to the formulas described in "Using the Encoder to Track Position" on page 81.

Syntax:           **?d or ?D**

Reply:            **nn**  
                    **mm**

Where:            **nn** = distance 1 value  
                    **mm** = distance 2 value

Notes:

The command returns a signed hexadecimal number where 0 to +127 is represented by 00 to 7F, and -1 to -127 is represented by FF to 80 respectively. The hexadecimal format is intended to be deciphered by a microcontroller. When exercising the controller manually, you may use the Decimal to Hexadecimal conversion table on page 167.

## Read Speed/Distance

Description:

This query will cause the controller to return either the speed or the distance computed by the Encoder module, depending on the operating mode that is selected. This command is similar to either of the two previous ones, except that it is read from a different location inside the controller and is a filtered value that smoothed abrupt changes.

Syntax:           **?k or K**

Reply:            **nn**  
                    **mm**

Where:            **nn** = speed 1 or distance 1 value  
                    **mm** = speed 2 or distance 2 value

## Read Encoder Limit Switch Status

Description:

This query will cause the controller to return the status of the four optional Encoder limit switches. The returned value is a two-digit (8-bit) Hexadecimal number of which each of the 4 least significant bits represents one of the hardware switches.

Note that for this function to work, limit switches must be connected to the encoder module using the special wiring diagram shown in "Wiring Optional Limit Switches" on page 78. If no limit switches are present, this query will return the logic levels of each of the encoder's quadrature outputs, which in most cases is not relevant information.

Syntax:           **?w or ?W**

Reply: **0n**

Where: **n** = switch status

The relationship between the value of n and the switch status is shown in the table below. Extracting the status of a given switch from this number is easily accomplished in software using masking.

TABLE 30. Reported value and switch status relationship

n Value	Switch				n Value	Switch			
	4	3	2	1		4	3	2	1
0	0	0	0	0	8	1	0	0	0
1	0	0	0	1	9	1	0	0	1
2	0	0	1	0	A	1	0	1	0
3	0	0	1	1	B	1	0	1	1
4	0	1	0	0	C	1	1	0	0
5	0	1	0	1	D	1	1	0	1
6	0	1	1	0	E	1	1	1	0
7	0	1	1	1	F	1	1	1	1

Note that the 0 and 1 levels represent a Closed Switch and Open Switch status, respectively.

## Read / Modify Encoder Module Registers and Parameters

### Description

These commands make it possible to examine and change parameters that are stored in the Encoder's module MCU RAM. While this command provides unrestricted access to up to 256 memory locations, a small number of these locations should never be read or altered. Parameter address and returned values are two digit Hexadecimal numbers (8-bit).

## Important Note

**Command character has been changed from "\$" to "\*" starting in version 1.7 of the controller firmware.**

### Read parameter

Syntax: **\*mm**

Reply: **DD**

Where **mm**= address location of parameter  
**DD**= parameter value

Example:

**\*84**                    Read value of parameter at address hex 84  
**01**                      Controller replies, value is 01

**Modify parameter**

Syntax:                **\*mm nn**

Reply:                 + if command was executed successfully  
                              - if error

Where                 **mm**= parameter address  
                              **nn**= new parameter value

Examples:

**\*84 03**                Store 03 into parameter at address hex 84

Notes:

All parameters and values are expressed with 2 hexadecimal digits.

The table below lists maps the few relevant parameters that can be accessed using this command.

TABLE 31. Encoder Registers

<b>Address</b>	<b>Parameter Description</b>	<b>Size</b>	<b>Access</b>
*84	Encoder Hardware ID Code	1 byte	Full
*85	Switches Status	1 byte	Full
*86	Speed or Distance 1 (depending on operating mode)	1 byte	Full
*87	Speed or Distance 2 (depending on operating mode)	1 byte	Full
*88	Counter Read/Write Mailbox MSB (bits 31 to 24)	4 bytes	Full
*89	Counter Read/Write Mailbox (bits 23 to 16)		
*8A	Counter Read/Write Mailbox (bits 15 to 8)		
*8B	Counter Read/Write Mailbox LSB (bits 7 to 0)		
*8C	Counter 1 MSB (bits 31 to 24)	4 bytes	Limited
*8D	Counter 1 (bits 23 to 16)		
*8E	Counter 1 (bits 15 to 8)		
*8F	Counter 1 LSB (bits 7 to 0)		
*90	Counter 2 MSB (bits 31 to 24)	4 bytes	Limited
*91	Counter 2 (bits 23 to 16)		
*92	Counter 2 (bits 15 to 8)		
*93	Counter 2 LSB (bits 7 to 0)		
*94	Destination Register 1 MSB (bits 31 to 24)	4 bytes	Full
*95	Destination Register 1 (bits 23 to 16)		
*96	Destination Register 1 (bits 15 to 8)		
*97	Destination Register 1 LSB (bits 7 to 0)		

TABLE 31. Encoder Registers

Address	Parameter Description	Size	Access
*98	Destination Register 1 MSB (bits 31 to 24)	4 bytes	Full
*99	Destination Register 1 (bits 23 to 16)		
*9A	Destination Register 1 (bits 15 to 8)		
*9B	Destination Register 1 LSB (bits 7 to 0)		
*9C	Distance 1 (when Position Mode enabled)	1 byte	Full
*8D	Distance 2 (when Position Mode enabled)	1 byte	Full
*86	Speed 1	1 byte	Full
*87	Speed 2	1 byte	Full
*A2	Time Base for speed computation of Encoder 1. Multiply this number by 256us to obtain the actual Time Base period.	1 byte	Full
*A3	Time Base for speed computation of Encoder 2. Multiply this number by 256us to obtain the actual Time Base period.	1 byte	Full
*A4	Encoder threshold level (see "Voltage Levels, Thresholds and Limit Switches" on page 76)	1 byte	Full
*A5	Distance divider	1 byte	Full
*A6	Mode Selection	1 byte	Limited

## Important Warning

**Do not alter any other area locations, as this may cause program execution failure inside the encoder module.**

## Register Description

### Encoder Hardware ID code

**Address: \*84**

Returns a 4-bit number identifying the encoder module hardware version and the status of two on-board jumpers. For Roboteq use only.

### Switch Status

**Address: \*85**

Returns a 4 bit number (4 least significant bits of the byte), each representing the state of one of the limit switches when installed. The ?W command described at "Read Encoder Limit Switch Status" on page 157 is a preferred method for reading this information.

### Speed or Distance 1 or 2

**Address: \*86 - Channel 1  
\*87 - Channel 2**



These two registers contain either the measured speed or the measured distance. Whether speed or distance information is returned depends on the settings contained in the Mode register described at. This information is returned using the ?p query (see "Query Analog Inputs" on page 140).

### Counter Read/Write Mailbox

**Address:**       **\*88 - Most Significant Byte**  
                  **\*89**  
                  **\*8A**  
                  **\*8B - Least Significant Byte**

Since the counters are 32-bits wide and accesses are 8-bit wide, it would normally take four separate accesses to fully read or write any of the counters. If the motors are running and the counter is changing in-between these accesses, inaccurate data will be either read or written. Therefore a two-step process is implemented for accessing the encoder's counters: for loading a new value in the counter, the value must first be loaded in the mailbox. It is then transferred in a single step using a command. When reading a counter, a read command is sent to the encoder module which then copies the counter value into the mailbox. The mailbox system can be used in the same way for reading and writing the destination register.

In practice, reading a counter is done by a single command described in "Read Encoder Counter" on page 155. This command will perform the steps above and output the selected counter value.

Writing a user-defined value into a counter or destination register requires that the value be loaded in the mailbox using the steps defined in "Read / Modify Encoder Module Registers and Parameters" on page 158, and then that of one of the commands described in "Set/Reset Encoder Counters and Destination Registers" on page 155 be issued.

### Counter 1 and 2

**Address:**       **\*8C - Most Significant Byte Counter 1**  
                  **\*8D**  
                  **\*8E**  
                  **\*8F - Least Significant Byte**  
  
                  **\*90 - Most Significant Byte Counter 2**  
                  **\*91**  
                  **\*92**  
                  **\*93 - Least Significant Byte**

These two 32-bit (4-bytes) registers are the actual counters. As discussed above, they should not be accessed directly, as their value may fluctuate between the four accesses needed to read or write a complete 32-bit counter.

### Destination Register 1 and 2

**Address:**       **\*94 - Most Significant Byte Destination 1**  
                  **\*95**  
                  **\*96**  
                  **\*97 - Least Significant Byte**  
  
                  **\*98 - Most Significant Byte Destination 2**  
                  **\*99**

**\*9A**

**\*9B - Least Significant Byte**

These two 32-bit (4-bytes) registers are used to store the desired destination when the controller is used in position mode. These registers should always be set using the mailbox mechanism described above. See "Using the Encoder to Track Position" on page 81 for a complete description of the position mode.

## **Distance 1 and 2**

**Address:**       **\*9C - Channel 1**  
                  **\*9D - Channel 2**

These registers contain a signed 8-bit value (-127 to +127) that represents the distance between the current counter position and the desired destination. This number is computed using a formula described in section "Using the Encoder to Track Position" on page 81.

## **Speed 1 and 2**

**Address:**       **\*9E - Channel 1**  
                  **\*8F - Channel 2**

These registers contain a signed 8-bit value (-127 to +127) that represents the motor speed relative to a maximum speed, which in turn depends on the number of encoder counts and time base settings as described in "Using the Encoder to Measure Speed" on page 80.

## **Time Base 1 and 2**

**Address:**       **\*A2 - Channel 1**  
                  **\*A3 - Channel 2**

These registers contain the timing information for measuring the speed. See "Using the Encoder Module to Measure Distance" on page 80 for a detailed description.

## **Encoder Threshold**

**Address:**       **\*A4**

This register contains a value that is used to detect a logic level 1 vs. a 0 at any of the 4 encoder input lines. The voltage threshold is computed as follows:

$$\text{Voltage Threshold} = 5V * \text{Register Value} / 255$$

See "Voltage Levels, Thresholds and Limit Switches" on page 76 for a detailed description.

## **Distance Divider**

**Address:**       **\*A5**

This registers contain the divider ratio that is applied to the difference between the current position and destination. See "Using the Encoder Module to Measure Distance" on page 80 for a detailed description.

## Counter Read Data Format

When receiving a counter read query, the encoder module will output the value of its 32-bit counter. If all 32-bit are sent, this would require 8 ASCII digits to represent the value.

A 32-bit counter can store over 2 billion counts in each direction. In practice, it will be rare that counts will be so large that only a partial number of the counter's bits will be significant at any given time.

In order to create a more efficient data stream on the controller's serial port, a simple compression technique is implemented. The scheme eliminates all of the counter's most significant bits if they are at 0 (for a positive count number) or F (for a negative count number).

For example, if the counter value is Hex 00000015, the value 15 will be returned after a counter query.

For negative numbers, a count value of -5 (which is FFFFFFFB in hex), the response to the query will be B.

To distinguish between positive and negative numbers, the Encoder module will insert a 0 ahead of any number string starting with a digit value higher than 7 (i.e. 8 to F) to signify that the number is positive. For negative numbers, the Encoder module will insert an F ahead of any number string starting with a digit value lower than 8 (i.e. 0 to 7). The table below shows examples of this scheme as applied to various counter values.

TABLE 32. Example counter values and RS232 output using a reduction scheme

<b>Decimal</b>	<b>32-bit Hex</b>	<b>Controller Output</b>
+5	00000005	5
+250	000000FA	0FA
-6	FFFFFFFA	A
-235	FFFFFF15	F15
+91,011,186	056CB872	56CB872
-7,986,091	FF862455	862455

When reading the counter value into a microcomputer, the reverse operation must take place: any output that is less than 8 digit long must be completed with a string of 0's if the first digit is of value 0 to 7, or with a string of F's if the first digit is of value 8 to F.

The resulting Hex representation of a signed 32-bit number must then be converted to binary or decimal as required by the application.

The burden of this extra processing is more than offset by the bandwidth relief on the controller's serial port.

## Encoder Testing and Setting Using the PC Utility

Extensive diagnostic, calibration, setting and testing support is provided in the Roborun PC utility. Basic instructions on how to install and run the PC utility can be found in "Using the Roborun Configuration Utility" on page 177



## Automatic Switching from RS232 to RC Mode

In many computer controlled applications, it may be useful to allow the controller to switch back to the RC mode. This would typically allow a user to take over the control of a robotic vehicle upon computer problem.

While the AX2550 can operate in either RC Radio or RS232 mode, the RS232 Data Input and RC Pulse Input 1 share the same pin on the connector. External hardware is, therefore, needed to switch this pin from the RS232 source or the RC Radio. The diagram in Figure 93 shows the external hardware required to perform such a switch.

A third RC channel is used to activate a dual-throw relay. When the radio is Off, or if it is On with the channel 3 off, the relay contact brings the RS232 signal to the shared input. The second relay contact maintains the Power Control wire floating so that the controller remains on.

When the RC channel 3 is activated, the relay turns On and brings the RC radio signal 1 to the shared input. The second relay contact brings a discharged capacitor onto the Power Control wire causing the controller to reset. Resetting the controller is necessary in order to revert the controller in the RC mode (the controller must be configured to default to RC mode).

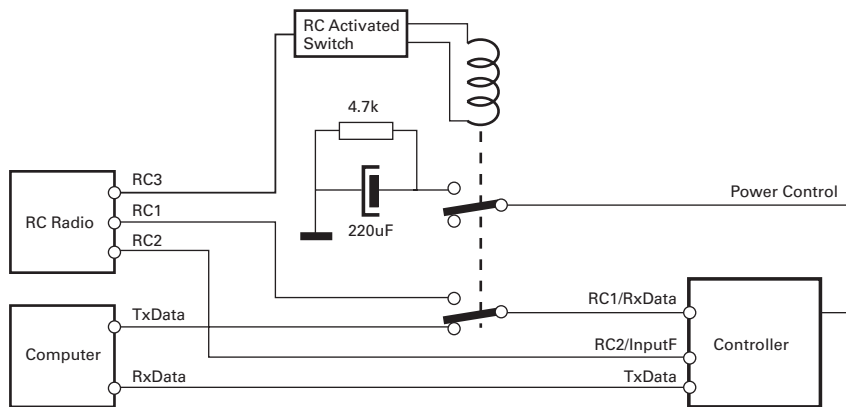


FIGURE 93. External circuit required for RS232 to RC switching

The switching sequence goes as follows:

Upon controller power on with Radio off: (or Radio on with RC ch3 off)

- Controller runs in RC mode (must be configured in RC mode)
- Computer must send 10 consecutive Carriage Returns. Controller enters RS232 mode

Controller is on, Radio turns On with RC ch3 On

- Controller is reset, returning to RC mode
- Controller will output the continuous parameter strings on the RS232 output. Computer thus knows that RC mode is currently active. Computer sends Carriage Return strings to try to switch controller back in RS232 mode. Since the RS232 line is not connected to the controller, mode will not change.

Controller is on, Radio is turned Off (or Radio On with RC ch3 Off)

- Relay deactivates. RS232 is now connected to shared input.
- String of Carriage Returns now received by controller.
- Computer looks for OK prompt to detect that the RS232 mode is now active.

Note: Wait 5 seconds for the capacitor to discharge before attempting to switch to RC mode if doing this repeatedly. Controller will not reset otherwise.

## Analog and R/C Modes Data Logging String Format

When the controller is configured in R/C or Analog mode, it will automatically and continuously send a string of ASCII characters on the RS232 output.

This feature makes it possible to log the controller's internal parameters while it is used in the actual application. The data may be captured using a PC connected via an RS232 cable

FIGURE 1. Pin connections to the controller's PDA installed in the actual robot. Details on how to wire the DB15 connector is described on page 122 for the R/C mode and on page 129 for the Analog mode.

This string is composed of a start character delimiter, followed by 12 or 13 two-digit Hexadecimal numbers representing 12 or 13 internal parameter values, and ending with a Carriage Return character. The figure below shows the structure of this string.

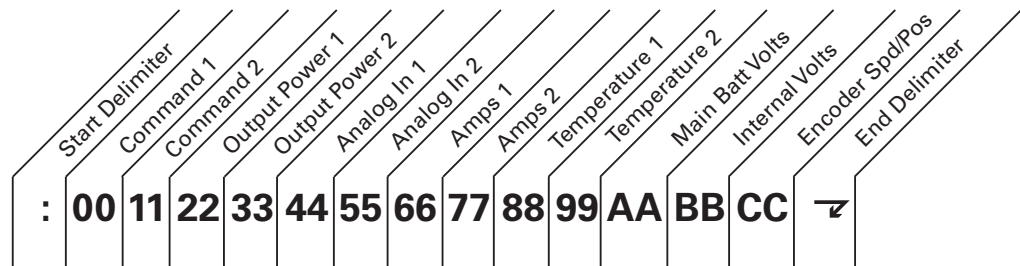


FIGURE 94. ASCII string sent by the controller while in R/C or Analog mode

The hexadecimal values and format for each parameter is the same as the response to RS232 queries described in page 138. The Encoder Speed/Position parameter is output only on the AX2850 or the AX2550 with Encoder module.

Characters are sent by the controller at the rate of one every 8ms. A complete string is sent in 213ms or 224ms.

## Data Logging Cables

The wiring diagrams shown in the figures below describe an easy-to-assemble cable assembly for use to create insertion points where to connect the PC for debug and data

logging purposes. This cable has a 15-pin male connector and 3 15-pin connectors. The

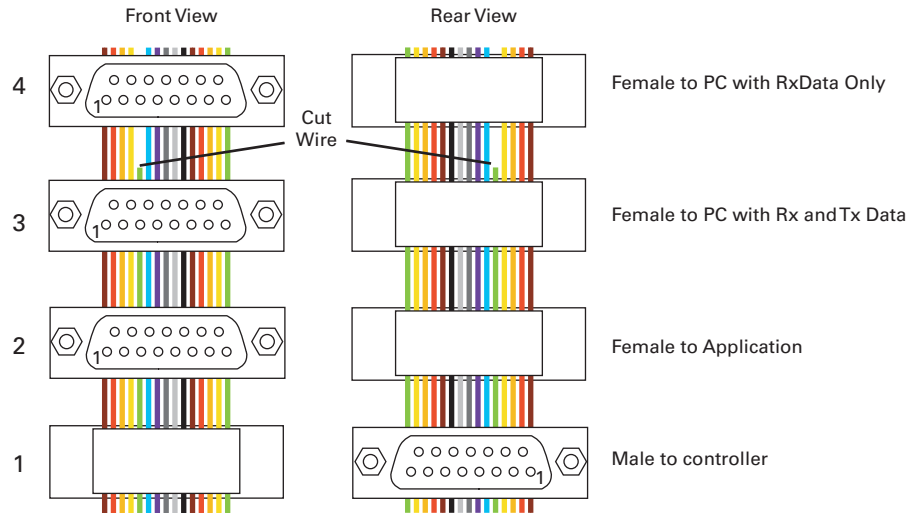


FIGURE 95. ASCII string sent by the controller while in R/C or Analog mode

male connector plugs into the controller. The application cable that would normally plug into the controller may now be plugged into one of the adapter’s female connector 2. The PC can be plugged into connector 3 or 4. Connector 3 has the Rx and Tx data lines needed for full duplex serial communication, thus allowing the PC to send commands to the controller. Connector 4 has the Rx line cut so that only a data flows only from the controller to the PC. This configuration is for capturing the data logging strings sent in the RC or Analog modes.

## Decimal to Hexadecimal Conversion Table

The AX2550 uses hexadecimal notation for accepting and responding to numerical commands. Hexadecimal is related to the binary system that is used at the very heart of micro-computers. Functions for converting from decimal to hexadecimal are readily available in high level languages such as C.

If the user intends to enter commands manually using the terminal emulation program, the conversion table in Table 33 can be used to do the translation. Note that the table only shows numbers for 0 to 127 decimal (00 to 7F hexadecimal). The AX2550’s speed commands are within this range. Table 34 shows the conversion values for numbers between 128 and 255 (unsigned) and between -1 and -128 (signed)

TABLE 33. 0 to +127 signed or unsigned decimal to hexadecimal conversion table

Dec	Hex	Dec	Hex	Dec	Hex	Dec	Hex
0	00	32	20	64	40	96	60
1	01	33	21	65	41	97	61
2	02	34	22	66	42	98	62
3	03	35	23	67	43	99	63
4	04	36	24	68	44	100	64

TABLE 33. 0 to +127 signed or unsigned decimal to hexadecimal conversion table

Dec	Hex	Dec	Hex	Dec	Hex	Dec	Hex
5	05	37	25	69	45	101	65
6	06	38	26	70	46	102	66
7	07	39	27	71	47	103	67
8	08	40	28	72	48	104	68
9	09	41	29	73	49	105	69
10	0A	42	2A	74	4A	106	6A
11	0B	43	2B	75	4B	107	6B
12	0C	44	2C	76	4C	108	6C
13	0D	45	2D	77	4D	109	6D
14	0E	46	2E	78	4E	110	6E
15	0F	47	2F	79	4F	111	6F
16	10	48	30	80	50	112	70
17	11	49	31	81	51	113	71
18	12	50	32	82	52	114	72
19	13	51	33	83	53	115	73
20	14	52	34	84	54	116	74
21	15	53	35	85	55	117	75
22	16	54	36	86	56	118	76
23	17	55	37	87	57	119	77
24	18	56	38	88	58	120	78
25	19	57	39	89	59	121	79
26	1A	58	3A	90	5A	122	7A
27	1B	59	3B	91	5B	123	7B
28	1C	60	3C	92	5C	124	7C
29	1D	61	3D	93	5D	125	7D
30	1E	62	3E	94	5E	126	7E
31	1F	63	3F	95	5F	127	7F

TABLE 34. +128 to 255 unsigned and -1 to -128 signed decimal to hexadecimal conversion table

UDec	Dec	Hex	UDec	Dec	Hex	UDec	Dec	Hex	UDec	Dec	Hex
-128	128	80	-96	160	A0	-64	192	C0	-32	224	E0
-127	129	81	-95	161	A1	-63	193	C1	-31	225	E1
-126	130	82	-94	162	A2	-62	194	C2	-30	226	E2
-125	131	83	-93	163	A3	-61	195	C3	-29	227	E3
-124	132	84	-92	164	A4	-60	196	C4	-28	228	E4
-123	133	85	-91	165	A5	-59	197	C5	-27	229	E5
-122	134	86	-90	166	A6	-58	198	C6	-26	230	E6



TABLE 34. +128 to 255 unsigned and -1 to -128 signed decimal to hexadecimal conversion table

<b>UDec</b>	<b>Dec</b>	<b>Hex</b>	<b>UDec</b>	<b>Dec</b>	<b>Hex</b>	<b>UDec</b>	<b>Dec</b>	<b>Hex</b>	<b>UDec</b>	<b>Dec</b>	<b>Hex</b>
-121	135	87	-89	167	A7	-57	199	C7	-25	231	E7
-120	136	88	-88	168	A8	-56	200	C8	-24	232	E8
-119	137	89	-87	169	A9	-55	201	C9	-23	233	E9
-118	138	8A	-86	170	AA	-54	202	CA	-22	234	EA
-117	139	8B	-85	171	AB	-53	203	CB	-21	235	EB
-116	140	8C	-84	172	AC	-52	204	CC	-20	236	EC
-115	141	8D	-83	173	AD	-51	205	CD	-19	237	ED
-114	142	8E	-82	174	AE	-50	206	CE	-18	238	EE
-113	143	8F	-81	175	AF	-49	207	CF	-17	239	EF
-112	144	90	-80	176	B0	-48	208	D0	-16	240	F0
-111	145	91	-79	177	B1	-47	209	D1	-15	241	F1
-110	146	92	-78	178	B2	-46	210	D2	-14	242	F2
-109	147	93	-77	179	B3	-45	211	D3	-13	243	F3
-108	148	94	-76	180	B4	-44	212	D4	-12	244	F4
-107	149	95	-75	181	B5	-43	213	D5	-11	245	F5
-106	150	96	-74	182	B6	-42	214	D6	-10	246	F6
-105	151	97	-73	183	B7	-41	215	D7	-9	247	F7
-104	152	98	-72	184	B8	-40	216	D8	-8	248	F8
-103	153	99	-71	185	B9	-39	217	D9	-7	249	F9
-102	154	9A	-70	186	BA	-38	218	DA	-6	250	FA
-101	155	9B	-69	187	BB	-37	219	DB	-5	251	FB
-100	156	9C	-68	188	BC	-36	220	DC	-4	252	FC
-99	157	9D	-67	189	BD	-35	221	DD	-3	253	FD
-98	158	9E	-66	190	BE	-34	222	DE	-2	254	FE
-97	159	9F	-65	191	BF	-33	223	DF	-1	255	FF



# **Configuring the Controller using the Switches**

---

The AX2550 Speed Controller can be programmed to operate in many modes using a simple set-up procedure. Two buttons and a 7-segment LED display allow the user to examine and change these settings. Alternatively, the controller may be programmed using a PC connected to the AX2550 controller through the RS232 serial communication port.

The new settings are then permanently stored in the controller's Flash memory so that jumpers are not required, resulting in improved system reliability.

---

## **Programming Methods**

There are three methods for programming the AX2550's settings:

- Using the controller's built-in switches and display. This method is described in details in this chapter.
- Using the PC-based Configuration Utility. See "Using the Roborun Configuration Utility" on page 177.
- Sending RS232 commands manually. See "Automatic Switching from RS232 to RC Mode" on page 165.

---

## **Programming using built-in Switches and Display**

Two switches and an LED display are provided to let you easily configure the controller in any of its many operating modes. Unlike the RS232 programming mode, the switches will let you configure the controller without the need for external hardware or special connectors. In this mode, the controller may be configured while installed on the robot without the need for special tools or a PC. Figure 96 shows the placement of the switches and display.

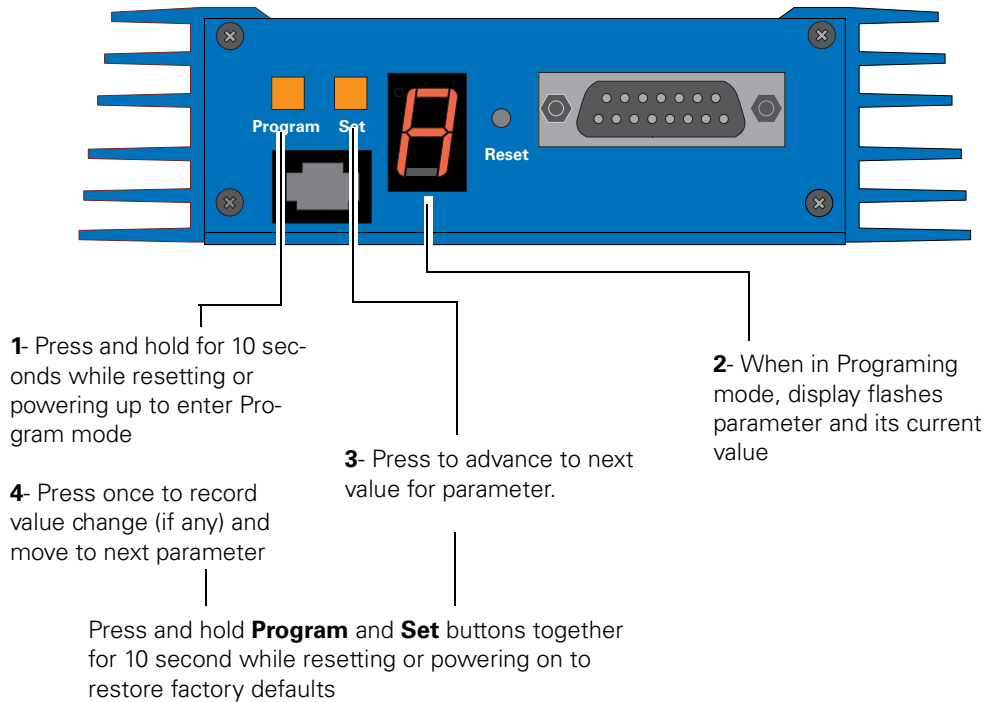


FIGURE 96. Operating the controller’s buttons and display

### Entering Programming Mode

Programming mode is entered by pressing and holding the **Program** button for 10 seconds after resetting the controller. The controller can be reset by powering it down and up or by pressing the **Reset** switch inserting a paper clip in the hole.

While the button is pressed and until the controller enters the programming mode, the display will show the following steady pattern.



After 10 seconds, the controller will enter the programming mode and flash a letter representing the first parameter in the list, followed by its numerical value.

### Important Warning

**Be careful not to confuse the Set and Program button when entering the Programming Mode. Pressing and holding the Set button alone for 10 seconds after reset will cause the controller to enter in self-test mode. This will cause the motors to be turned On and Off according the test sequence described in the Self Test section of this manual. See "Self-Test Mode" on page 53.**

## Changing parameters

Pressing the Set button while in Programming Mode will cause the value of the parameter being inspected to be incremented by one. When the maximum valid value is reached, pressing Set again will cause the value to restart at 0.

When the desired value is displayed, press the **Program** button to store it in the controller's non-volatile memory. This will also cause the controller to display the next parameter and its current value.

**Note** that a new parameter value will **ONLY** be saved if the **Program** button is pressed after the value has been changed with the **Set** button.

Additionally, once you have pressed the **Set** button and begin changing a parameter's value, it is not possible to cancel the change. If you wish to leave a parameter value unchanged after you have started changing it, you must press the **Set** button again several times until it goes back to its original value. Alternatively, you can reset or power Off the controller to prevent the new value to be recorded.

## The Special Case of Joystick Calibration

When the Joystick parameter is selected ("J" and "-" flashing), pressing the **Set** key again will cause the controller to enter the Radio Control Joystick calibration sequence.

Once the joystick calibration mode is entered, the display will flash "J" and "o". With the R/C receiver and transmitter On, move both joysticks to their full forward, full back, full left and full right positions a few times. This will cause the controller to capture the min. and max. joystick position values.

Then move the joysticks to their central (rest) positions. Press the **Program** button to save these values in the controller's Flash memory. The min. and max. values saved are these captured when the joystick was moved around. The center values saved are the position of the joystick as it was when **Program** was pressed.

**Note** that once the Joystick calibration mode is entered, you must go through the complete calibration sequence described above. If the joysticks are not moved, incoherent data may be saved in the Flash causing the controller to malfunction in the R/C mode. If bad calibration data is saved in the Flash, try calibrating again by repeating the entire Joystick Programming sequence, otherwise you may restore the factory defaults (this will cause all parameters you have changed to be restored to their default values as well)

The Joystick calibration operation is fully described in the R/C mode chapter. See "Left / Right Tuning Adjustment" on page 49.

## Restoring factory defaults

Should you, for any reason, require to reset the AX2550 controller to its factory default value, press and hold the Program and Set button together for 10 seconds while resetting

the controller. All parameters, including Joystick limits, will be reset to their default values shown in the “Programmable Parameters List” on page 174

### **Exiting the Parameter Setting Mode**

Exiting the Programming Mode can be done only by pressing the **Reset** button or powering down the controller. The new parameters will be the ones in use after the controller is reset or first powered up.

---

### **Programmable Parameters List**

The following table shows the AX2550’s controller parameters in the order they appear during programming, as well as their valid values.

### **Important Notice**

**The parameter table below is guaranteed to be accurate only if the controller software version number matches the one of this manual. See “Obtaining the Controller’s Software Revision Number” on page 23 for instructions on how to find this number.**

**This manual is for software version 1.9b**

If the controller has a more current software revision, please download an updated version of this manual from the Roboteq web site at [www.Roboteq.com](http://www.Roboteq.com).

**This table shows only the parameters that can be programmed using the switches and display.** Other less commonly used parameters exist and are only accessible and programmable using the Configuration Utility (page 177) or the RS232 commands (page 143).

TABLE 35. Parameters accessible using the controller's switches and display

<b>Order</b>	<b>Letter</b>	<b>Description</b>	<b>Possible Values (default)</b>	<b>See pages</b>
1	I	Input Command mode:	(0) = R/C Radio mode (default) 1 = RS232 full duplex, no watchdog 2 = RS232 half duplex, with watchdog 3 = Analog mode	page 109 page 131 page 123
2	C	Motor Control mode	(0) = Separate A, B, speed control, open loop (default) 1 = Mixed A & B, speed control, open loop <b>For safety reasons, the modes below cannot be selected using the switches.</b> 2 = Speed control on A, open loop. Position control on B 3 = A & B Position control 4 = Separate A, B, speed control, closed loop 5 = Mixed A & B, speed control, closed loop 6 = Speed control on A, closed loop. Position control on B	page 42
3	A	Amp limit	0 = 30A 1 = 45A 2 = 60A 3 = 75A 4 = 90A (5) = 105A default 6 = 120A Amps may be set with a finer resolution using the PC utility	page 44
4	S	Acceleration	0 = very slow 1 = slow (2) = medium (default) 3 = medium 4 = fast 5 = fastest Acceleration may be set with a finer resolution using the PC utility	page 47

TABLE 35. Parameters accessible using the controller's switches and display

Order	Letter	Description	Possible Values (default)	See pages
5	U	Input switch function	0 = causes emergency stop 1 = invert commands switch (2) = no action (default)	page 51 page 52
6	b	Brake/Coast	Not implemented	
7	d	R/C Joystick Dead-band Analog Input Dead-band	0 = no deadband *** 1 = 8% (2) = 16% (default) 3 = 24% 4 = 32% 5 = 40% 6 = 46% 7 = 54%	page 117 page 127
8	E	Exponentiation on channel 1	(0) = Linear (no exponentiation - default) 1 = strong exponential 2 = normal exponential 3 = normal logarithmic 4 = strong logarithmic	page 118
9	F	Exponentiation on channel 2	Same as E, above	
10	L	Left / Right Adjust	(7) = no adjustment (default) 0, 1, ..., 6 = -5.25%, -4.5%, ..., -0.75% 8, ..., D, E** = +0.75, ..., +4.5%, +5.25%	page 49
11	J	Joystick calibration	- = not calibrating o = in calibration mode	page 118

\*The coast function is not implemented in this revision of the software. The controller will only operate in brake mode.

\*\* Values are in hexadecimal numbers where the decimal values 10, 11, 12... 15 are represented with the letters A, B, C... F.

\*\*\* Deadband percent values shown are for R/C mode. For analog deadband values, see page 127.



# Using the Roborun Configuration Utility

---

A PC-based Configuration Utility is available, free of charge, from Roboteq. This program makes configuring and operating the AX2550 much more intuitive by using pull-down menus, buttons and sliders. The utility can also be used to update the controller's software in the field as described in "Updating the Controller's Software" on page 194.

---

## System Requirements

To run the utility, the following is need:

- PC compatible computer running Windows 98, Me, 2000, XP or Vista
- An unused serial communication port on the computer with a 9-pin, female connector.
- An Internet connection for downloading the latest version of the Roborun Utility or the Controller's Software
- 5 Megabytes of free disk space

If there is no free serial port available, the Configuration Utility can still run, but it will not be able to communicate with the controller.

If the PC is not equipped with an RS232 serial port, one may be added using an USB to RS232 converter.

---

## Downloading and Installing the Utility

The Configuration Utility is included on the CD that is delivered with the controller or may be obtained from the download page on Roboteq's web site at [www.roboteq.com](http://www.roboteq.com). It is recommended that you use the downloaded version to be sure that you have the latest update.

- download and run the file `robosetup.exe`
- follow the instructions displayed on the screen

- after the installation is complete, run the program from your Start Menu > Programs > Roboteq

The controller does not need to be connected to the PC to start the Utility.

---

## Connecting the Controller to the PC

The controller must be connected to the PC to use the Utility to perform any of the following functions:

- to read the current parameters stored in the controller and display them on the computer
- to store new parameters in the controller
- to exercise the motors using your PC
- to update the controller's software
- calibrate the Amps sensor

If the controller is not connected, the Configuration Utility can run and be used to automatically generate the setting codes for manual entry. See "Encoder Setting and Testing" on page 184.

Most computers have at least one, but often times two, serial ports. Look for one or two connectors resembling the illustration in Figure 97.

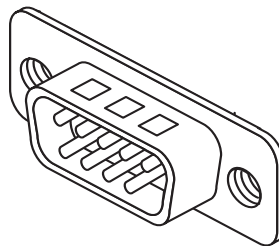


FIGURE 97. Look for a 9-pin male connector on your PC

If a serial port connector is already connected to something else, it may be possible to unplug the current device and temporarily connect the controller as long as the software operating the current device is not running. If no serial port is available on your PC, use an USB to RS232 adapter.

Connect the provided serial cable to the controller on one end and to the PC on the other.

Power the controller, preferably using the yellow Power Control wire, with a 12 to 40V battery or power supply with 200mA of minimum output.

Connect the Controller's Ground to the negative (-) terminal, and the Power Control input to the positive (+) of the power supply. The controller will turn On. If it doesn't, verify that the polarity is not reversed. Leave VMot unconnected unless you want to exercise the Motors.

Upon powering On, the controller will display "no ctrl" if configured in the R/C mode or a steady pattern if configured in the RS232 mode.

## Roborun Frame, Tab and Menu Descriptions

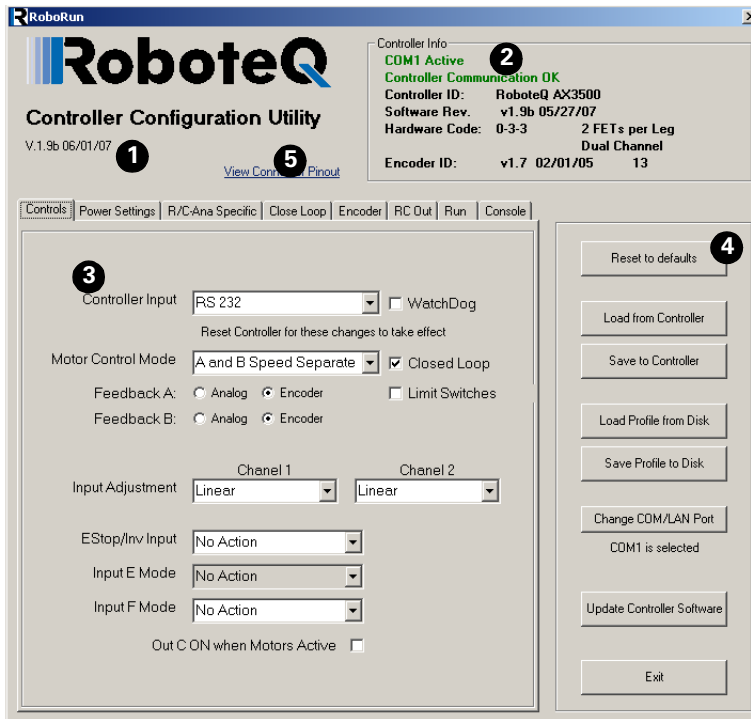


FIGURE 98. Roborun screen layout

The Roborun screen contains the four main set of commands and information frames described below:

### 1- Program Revision Number

This is the revision and date of the Roborun utility. It is recommended that you always verify that you have the latest revision of the utility from Roboteq's web site at [www.roboteq.com](http://www.roboteq.com)

### 2- Controller and Communication Link Information

This frame will automatically be updated with an indication that a free communication port was found and opened by the utility.

If no free communication port is available on your computer, it will be indicated in this window. Try to select another port using the "Change COM Port" button or try to free the port if it is used by a different device and program.

With the port open, Roborun will try to establish communication with the controller. If successful, this window will display the software revision, the revision date and a set of digits identifying hardware revision of the board inside the controller. New version of the AX2550

featuring the additional Analog Inputs 3 and 4 are automatically identified as Rev. B in this area of the screen.

### 3- Parameter Selection and Setting and Special Functions

This is the program's main frame and includes several types of tabs, each of which has several buttons, menus and other User Interface objects. These tabs and the functions they contain are described in detail in the following sections.

Navigate from one set of commands to another by clicking on the desired tab.

### 4- File and Program Management Commands

This frame contains a variety of buttons needed to load and save the parameters from and to the controller or disk. This frame also contains the button needed to initiate a software update to the controller.

### 5- View Controller Connector Pinout

Clicking on this link will conveniently pop a window containing the Controller's connector pinout.

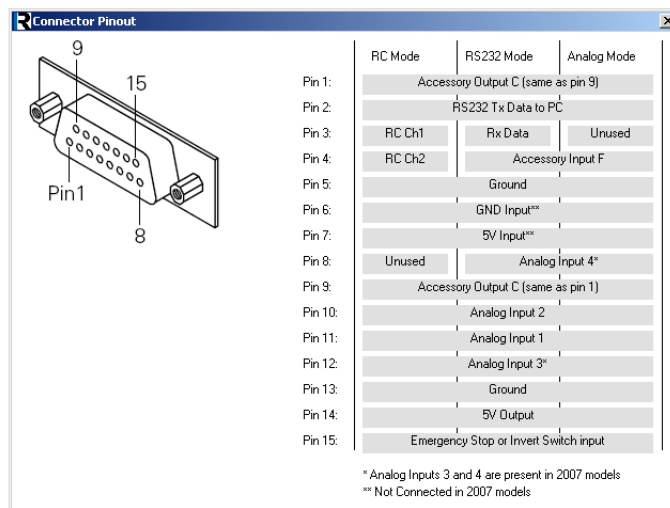


FIGURE 99. Roborun screen layout

## Getting On-Screen Help

The Roborun buttons and fields are very intuitive and self-explanatory. Additional explanations and help is provided by means of ToolTips for several of command. Simply move the cursor to a button, tab or other gadget on the screen and a message box will appear after a few seconds.

## Loading, Changing Controller Parameters

The first set of tabs allows you to view and change the controller’s parameters. These tabs are grouped according to the general type of parameters (Controls, Power Setting, and R/C Settings).

When starting Roborun, this screen is filled with the default values. If the controller is connected to your PC, Roborun will automatically detect it and ask you if you wish to read its settings.

The controller’s setting in the PC at can be read any other time by pressing the “Load from Controller” button. After changing a parameter, you must save it to the controller manually by pressing the “Save to Controller” button.

## Control Settings

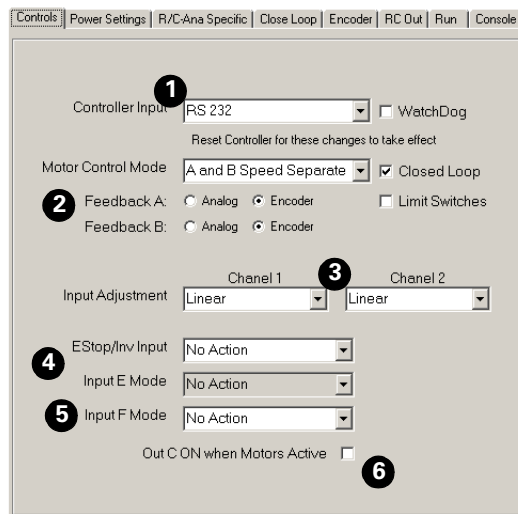


FIGURE 100. Control modes setting screen

The screen shown in Figure 100 is used to view and change the controller’s main control modes. Below is the list of the parameters accessible from this screen:

### 1- Controller Input:

This pull down menu allows the user to select the RS232, R/C or Analog mode of operation. If the RS232 mode is selected, a check box will appear, allowing you to enable or disable the RS232 Watchdog. For more information on these modes, see

- “R/C Operation” on page 109
- “Serial (RS-232) Controls and Operation” on page 131
- “RS-232 Watchdog” on page 138
- “Analog Control and Operation” on page 123

### 2- Motor Control Mode

This pull down menu is used to choose whether the controller will operate in Separate or Mixed mode. For more information on these modes, see “Selecting the Motor Control Modes” on page 42.

### 3- Input Command Adjustment

These pull down menus will let you select one of five conversion curves on each of the input command values. See “Command Control Curves” on page 48.

### 4- Emergency Stop or Invert Switch Select

This pull down menu allows the selection of the controller’s response to changes on the optional switch input: Emergency Stop, Invert Commands, or no action. See “Emergency Stop using External Switch” on page 51 and “Inverted Operation” on page 52.

### 5- Effect of Digital Inputs

This pull down menu allows the selection of the controller’s response to changes on either of the two digital inputs. See “Special Use of Accessory Digital Inputs” on page 52.

### 6- Output C Activation

This check box will cause the controller to activate when power is applied to one or both motors. See “Activating Brake Release or Separate Motor Excitation” on page 51.

## Power Settings

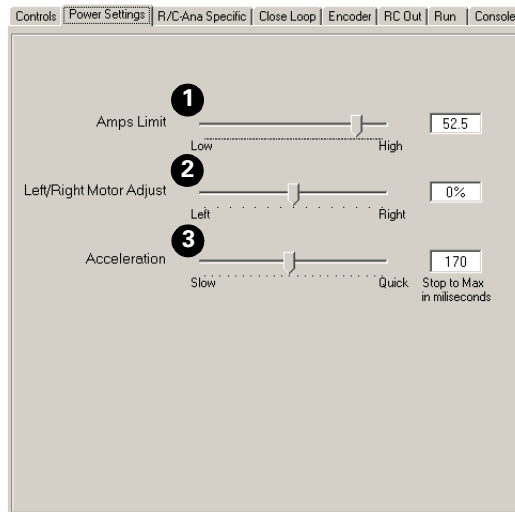


FIGURE 101. Power settings screen

The screen shown in Figure 101 is used to view and change the power parameters of the controller.

#### 1- Amps limit

This slider will let you select the max amps that the controller will deliver to the motor before the current limitation circuit is activated. See “User Selected Current Limit Settings” on page 44. Note that this limits the current flowing from the battery. The current flowing through the motor may be higher. See “Battery Current vs. Motor Current” on page 45.

### 2- Left/Right Adjust

This slider will let you configure the controller so that it applies more power to the motors in one direction than in the other. See “Left / Right Tuning Adjustment” on page 49.

### 3- Acceleration Setting

This slider will let you select one of seven preset acceleration values. The label on the right shows a numerical value which represents the amount of time the controller will take to accelerate a motor from idle to maximum speed. See “Programmable Acceleration” on page 47.

## Analog or R/C Specific Settings

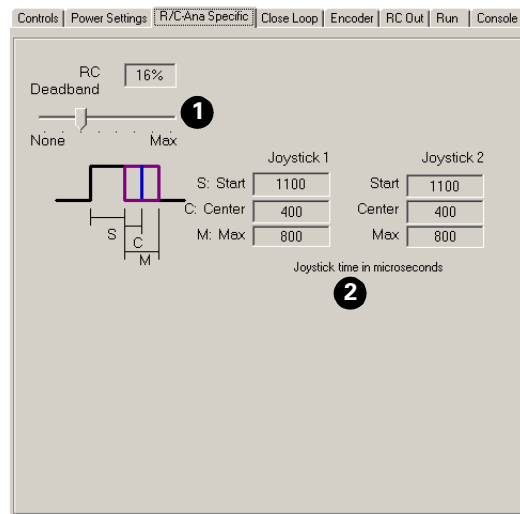


FIGURE 102. Power settings screen

The screen shown in Figure 102 slightly changes in function of whether or not the Analog Input mode is selected.

If the Analog Input mode is selected on the main screen, then this page is used to set the Analog Deadband value. In the R/C mode, this page is used to view and change parameters used in the R/C mode of operation. None of these parameters has any effect when running the controller in RS232 mode.

If the controller is configured in RS232 mode, some of these menus will turn gray but will remain active.

#### 1- Deadband

This slider will let you set the amount of joystick motion off its center position before the motors start moving. The slider will work identically in the R/C or analog mode, however, the % value will be different. See “Joystick Deadband Programming” on page 117 and “Analog Deadband Adjustment” on page 127.

#### 2- Joystick Timing

These fields are enabled only if the R/C mode is selected. These number areas will let you read and modify the R/C pulse timing information used by the controller. New values can be entered manually to create different capture characteristics. They are also useful for viewing the stored values after an automatic joystick calibration sequence. See “Joystick Calibration” on page 118 and “Automatic Joystick Calibration” on page 119.

## Closed Loop Parameters

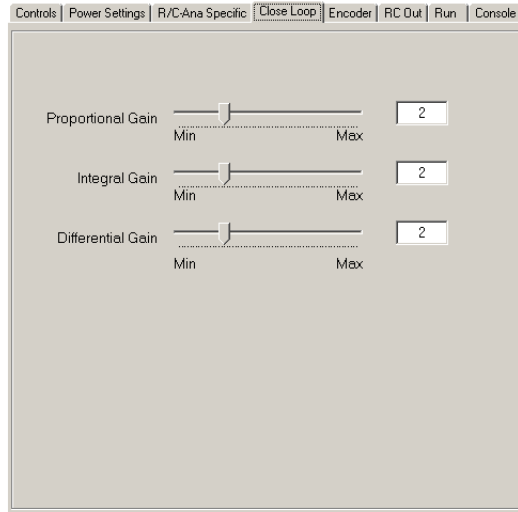


FIGURE 103. Closed Loop parameter setting screen

The screen shown in Figure 103 is used to set the Proportional, Integral and Differential gains needed for the PID algorithm. These PID gains are loaded after reset and apply to both channels. Gains can be changed individually for each channels and on-the-fly using RS232 commands. These parameters are used in the Position mode (see page 85) and the Closed Loop speed mode (see page 97).

## Encoder Setting and Testing

Extensive diagnostic, calibration, setting and testing support is provided in the Roborun PC utility. Basic instructions on how to install and run the PC utility can be found in “Using the Roborun Configuration Utility” on page 177.

Once the utility is up and running and the controller found and identified, click on the “Encoder” tab to bring up the Encoder configuration and setup screen show in Figure 104 below.



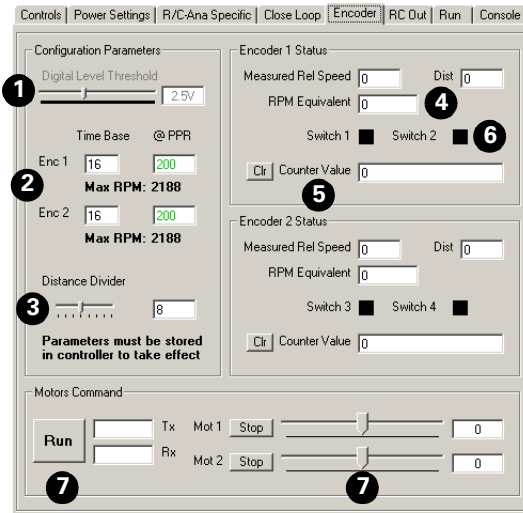


FIGURE 104. Encoder setup and test screen on Roborun

With this utility, the following actions can be accomplished:

- Set and program the Encoder module's parameters in EEPROM
- Activate the motors in each direction at variable speed
- View the measured encoder counts
- View the measured encoder speed
- View the status of the Limit Switches

The screen is composed of the following buttons and displays:

- 1- Setting of the Encoder's threshold level
- 2- Setting of the Time Base for speed computation
- 3- Setting Divider for computing relative distance
- 4- Measure and display speed and relative distance
- 5- Measure and display counter values
- 6- Detect and display optional limit switch status
- 7- Start/Stop communication with controller
- 8- Set motor speed and direction for testing

## Encoder Module Parameters Setting

The Encoder module has four programmable parameters: Two Time Bases (one for each encoder), a Divider for computing relative distance, and the voltage threshold for discerning a 0 or 1 level at the encoder's output. In the case of the AX2550, the threshold is fixed at 2.5V and cannot be changed.

The Time Base parameter is used to compute the speed measured by the module. The measured speed is a relative number ranging from 0 to +/-127.

The relationship between this relative speed number and the actual RPM is based on the Time Base value and the Encoder's Pulses Per Revolution (PPR) value (see "Using the Encoder to Measure Speed" on page 80 for details)

On this screen, changing the Time Base and PPR values automatically display the "Max RPM" values that can be measured with these settings. For example, with a default setting of 16 and 200 for the Time Base and PPR respectively, the maximum RPM values is 2188. This means that when the motors rotate at 2188 RPM, the measured relative speed is +127. If the motor spins faster, the speed is still reported to be +127.

Note that the PPR value is not stored in the controller. It is used only in Roborun to convert relative speed into actual RPM.

The Divider parameter is described in "Using the Encoder to Track Position" on page 81.

The threshold level parameter and its operation is described in "Voltage Levels, Thresholds and Limit Switches" on page 76.

## Exercising the Motors

A set of buttons and sliders is provided to start/stop communication with the controller and encoder.

When communication is active, the screen will be updated with Encoder data.

Moving the motor sliders will set the motors to the desired speed and direction.

## Viewing Encoder Data

During operation, this screen will display the following information:

- The instantaneous relative 0 to +/-127 speed value
- The instantaneous relative distance to destination (0 after reset)
- The actual speed computed from the measured relative speed value, encoder Time Base, and Encoder PPR. The **PPR value must be entered manually on this screen every the utility is run**, as it is not stored in the controller or on the PC.
- The Encoder counter values
- The status of the optional limit switches

---

## Running the Motors

The Roborun utility will let you exercise and monitor the motors, sensors and actuators using a computer. This feature is particularly useful during development as you will be able to visualize, in real-time, the robot's Amps consumption and other vital statistics during actual operating conditions.

Figure 105 shows the Run Screen and its numerous buttons and dials.

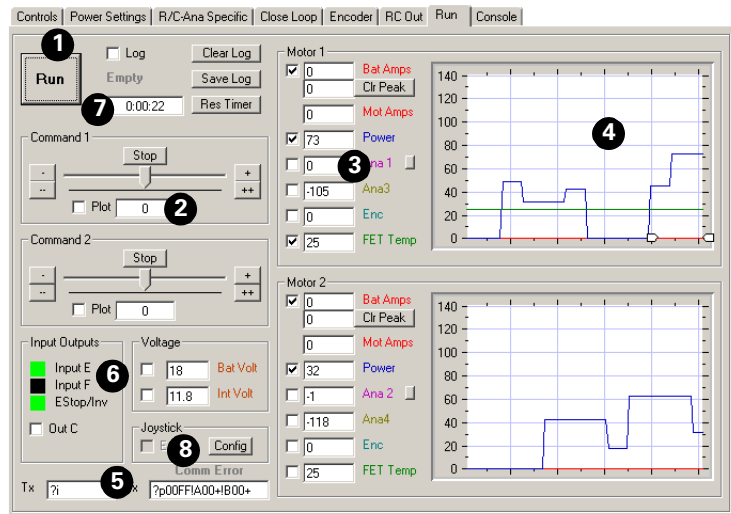


FIGURE 105. Motor exercising and monitoring screen

### 1- Run/Stop Button

This button will cause the PC to send the run commands to the controller and will update the screen with measurements received from the controller.

When the program is running, the button's caption changes to "Stop". Pressing it again will stop the motors and halt the exchange of data between the PC and the controller.

If another tab is selected while the program is running, the program will stop as if the Stop button was pressed.

### 2- Motor Power setting

This sub-frame contains a slider and several buttons. Moving the slider in any direction away from the middle (stop) position will cause a power command to be issued to the controller. The value of the command is shown in the text field below the slider.

The stop button will cause the slider to return to the middle (stop) position and a 0-value command to be sent to the controller. The + and ++ buttons will cause the slider to move by 1 or 10 power positions respectively.

### 3- Measurement

These series of fields display the various operating parameters reported in real-time by the controller:

The **Amps** field reports the current measured at each channel. The **Peak Amps** field will store the highest measured Amp value from the moment the program began or from the time at which the peak was reset using the Clr Peak button. **Motor Amps** is a calculated estimated value based on the batter amps and the current power level. See "Battery Current vs. Motor Current" on page 45.

The **Power** field displays the power level that is actually being applied to the motor. This value is directly related to the motor command except during current limitation, in which

case the power level will be the one needed to keep the Amps within the limit. Note that the display value is not signed and thus does not provide rotation direction information.

The **Ana** fields contain the analog input values that are measured and reported by the controller. When the controller is in the position mode with Analog Feedback, the Ana1 and Ana2 fields will display the position sensed on the feedback potentiometer. When in speed mode, these fields display the measured speed by the tachometers. When the controller is in Analog command mode, the Ana 1 and Ana 2 show the value of the command potentiometer, while the feedback is on Ana 3 and Ana 4.

In all other modes, this field will display the value at the analog input pin. A small button next to this field will toggle the display caption, and change the conversion algorithm from raw analog, to volts or temperature.

Note that in order to measure and display the external temperature or voltage, the proper external components must be added to the controller. See "Connecting External Thermistor to Analog Inputs" on page 66 and "Using the Analog Inputs to Monitor External Voltages" on page 67.

The **Enc** field contains the speed or position (depending on the selected operating mode) measured by the Optical Encoder if enabled.

The **Temp** field displays the heat sink temperature for each channel

The **Bat Volt** field displays the main battery's voltage (voltage applied to the thick red wires).

The **Ctrlr Volt** field displays the controller's internal regulated 12V voltage.

#### 4- Real-Time Strip Chart Recorder

This chart will plot the actual Amps consumption and other parameters as measured from the controller. When active, the chart will show measurement during the last five seconds.

Traces for most parameters can be displayed or hidden by clicking on the checkboxes found next to their numeric fields.

#### 5- Transmit and Receive Data

These two fields show the data being exchanged between the PC and the controller. While these fields are updated too fast to be read by a person, they can be used to verify that a dialog is indeed taking place between the two units.

After the Start button is pressed, the Tx field will show a continuous string of commands and queries sent to the controller.

The Rx field will display the responses sent by the controller. If this field remains blank or is not changing even though the Tx field shows that data is being sent, then the controller is Off or possibly defective. Try resetting the controller and pressing the Run/Stop button.

These two fields are provided for quick diagnostic. Use the Console Tab for full visibility on the data exchange between the controller and the PC.

#### 6- Input Status and Output Setting

This section includes two check boxes and three color squares. The check marks are used to activate either of the controller's two outputs. The color blocks reflect the status of the

three digital inputs present on the controller. Black represents a “0” level. Green represents a “1” level.

### **7- Data Logging and Timer**

A timer is provided to keep track of time while running the motors. An additional set of buttons and displays are provided to operate a data logger. The data logger is fully described in the section that follows.

### **8- Joystick Enable**

Enable and configure a joystick.

## **Logging Data to Disk**

While running the motors, it is possible to have Roborun capture all the parameters that were displayed on the various fields and charts and save them to disk. The log function is capable of recording 32,000 complete sets of parameters, which adds up to approximately 30 minutes of recording time. The figure below details the buttons and check boxes needed to operate this function.

### **1- Log Check Box**

When checked, Roborun will capture all the parameters and save them in local RAM. The data is not saved to disk until the “Save to Disk” button is pressed. Data is being captured for as long as the program is in the Run mode, whether or not a motor command is applied.

### **2- Clear Log**

This button can be pressed at any time to clear the local RAM from its content. Clearing the log also has the effect of resetting the timer.

### **3- Log Fill Status**

This gray text box indicates whether the local RAM log is empty, full or in-between.

### **4- Reset Timer button**

The timer automatically runs when the Run button is pressed and data is being exchanged with the controller, regardless of whether or not logging is activated. This button resets the timer.

### **5- Save Log to Disk button**

Pressing this button will prompt the user to select a filename and location where to copy the logged data. The file format is a regular text file with each parameter saved one after the other, separated by a comma. The file extension automatically defaults to .csv (comma separated values) so that the data can be imported directly into Microsoft Excel. The first

line of the save file contains the Header names. Each following line contains a complete set of parameters. The Header name, order and parameter definition is shown in Table 36:

TABLE 36. Logged parameters order, type and definition

Parameter Header	Data type/range	Measured Parameter
Seconds	Integer	Timer value expressed in seconds
Command1	-127 to +127	Command applied to channel 1
Command2	-127 to +127	Command applied to channel 2
Power1	0 to 127	Amount of power applied to the output stage of channel 1
Power2	0 to 127	Same for channel 2
Ana 1, Speed 1, Pos 1 or Temp 1 or Volt 1	-127 to + 127 -40 to +150 0 to 55	Value of sensor connected on analog input 1. Data is automatically converted to the right value and format by Roborun according to the sensor that is being used.
Ana 2, Speed 2, Pos 2 Temp 2 or Volt 2	-127 to + 127 -40 to +150 0 to 55	Same for channel 2
Amps1	0 to 255	Measured Amps on channel 1
Amps2	0 to 255	Measured Amps on channel 2
FET Temp1	-40 to +150	Measured Temperature on channel 1's heatsink.
FET Temp2	-40 to +150	Measured Temperature on channel 2's heatsink.
Batt Volt	0 to 55	Main Battery Voltage.
Ctrl Volt	0 to 28.5	Internal 12V Voltage.
Enc1	-127 to + 127	Measured Optical Encoder's Speed or Position depending on selected operating mode
Enc2	-127 to + 127	Same for channel 2

## Connecting a Joystick

Exercising the motors can easily be done using a Joystick in addition to the on-screen sliders. Simply connect a joystick to the PC and enable it by clicking in the Joystick check box in the PC utility.

If the box is grayed out, the joystick is not properly installed in the PC. Click on the "Config Joystick" button to open a configuration screen and the joystick control panel.

Joystick movement should automatically translate into Channel 1 and Channel 2 command values and make the sliders move. These commands are also sent to the controller. In the Config Joystick panel, the Joystick may be configured so that the X-Y channels are swapped and the direction for each axis reversed.

It is strongly recommended that an USB rather than Analog joystick be used.

A joystick test program name “Joytest” is automatically installed in the Start menu when installing the Roborun utility. This program may be used to further verify that the joystick is properly installed in the PC and is fully operational.

## Using the Console

The console screen allows you to communicate with the controller using raw ASCII data. This function is very useful for troubleshooting when normal communication with Roborun cannot be established (e.g. “Controller not found”; no response to command changes, communication errors, ...etc.). The Roborun utility will let you exercise and monitor the motors, sensors and actuators using a computer. This feature is particularly useful during development as you will be able to visualize, in real-time, the robot’s Amps consumption and other vital statistics during actual operating conditions.

Figure 105 shows the Console Screen and its various components.

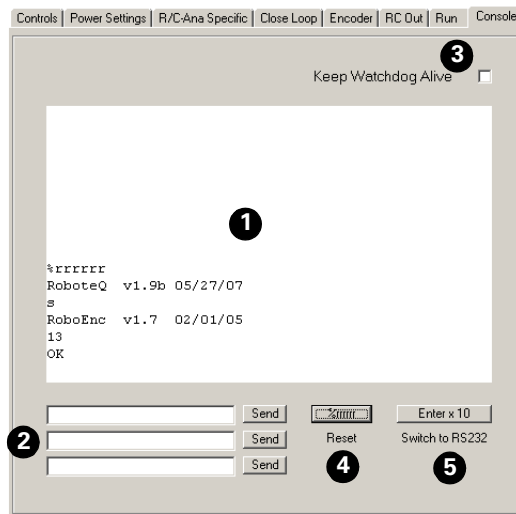


FIGURE 106. Raw ASCII data exchange in Console

### 1- Terminal Screen

This area displays the raw ASCII data as it comes out of the controller. After the controller is reset, it will output a prompt with the firmware’s revision and date. Then, if in the RC or Analog mode, the controller will output a continuous string of characters for data logging. If in RS232 mode, the controller will output an “OK” prompt and is ready to accept commands.

### 2- Command Entry

This window is used to prepare up to 3 command string and send them by clicking on their associated buttons. The string is sent to the controller when clicking on the send button. Commands can only be sent when the controller has entered in RS232 mode. See “Controller Commands and Queries” on page 138 for the complete list of commands and que-

ries. See “RS232 Encoder Command Set” on page 155 for the list of Encoder related commands and queries.

### 3- Keep Watchdog Alive

If the controller is in the RS232 mode with the watchdog enabled, then after 1 second of inactivity motors will be stopped if they were one and a “W” character will be sent to the terminal. When this checkbox is checked, Roborun will send a Null character to the controller on a regular basis so that the watchdog time-out is never reached.

### 4- Send Reset String

Clicking this button while the controller is in RS232 mode, will cause the reset string to be sent to the controller.

### 5- Send 10 Carriage Returns

Clicking this button will cause Roborun to send ten consecutive “Carriage Return” character. If the controller is configured in Analog or RC mode, the Carriage Returns will cause it to switch to RS232 mode until the controller is reset again.

---

## Viewing and Logging Data in Analog and R/C Modes

When the controller is configured in R/C or Analog mode, it will automatically and continuously send a string of ASCII characters on the RS232 output. “Analog and R/C Modes Data Logging String Format” on page 166 shows the nature and format of this data.

This feature makes it possible to view and log the controller’s internal parameters while it is used in the actual application. The data may be captured using a PC connected to the controller via an RS232 cable or wireless modem.

When wired for R/C or Analog controls, the AX2550 will not be able to receive commands from the PC and the Roborun software will not recognize the controller as being present. However, when in the Run tab and the Run button activated, Roborun will be receiving the strings sent by the controller and display the various parameters in the right display box and chart.

---

## Loading and Saving Profiles to Disk

It is possible to save the configuration parameters that are read from the controller or that have been set/changed using the various menus to the disk. This function allows easy recall of various operating profiles at a later time without having to remember or manually reset all the parameters that are used from one configuration to another.

To save a profile to disk, simply click on the “Save Profile to Disk” button. You will then be prompted to choose a file name and save.

Reading a profile from disk is as simple as clicking on the “Load Profile from Disk” button and selecting the desired profile file. The parameters will be loaded in each of their respective buttons, sliders and text fields on the various Roborun screens. **The parameter will not be transferred to the controller until you press the “Save to Controller” button.**



## Operating the AX2550 over a Wired or Wireless LAN

The Roborun utility supports connection and operation of the AX2550 controller over a Wired or Wireless TCP/IP network. This feature makes it easy to tele-operate and monitor the controller across a lab or across the globe via Internet.

To operate over the network, two computers are required, as show in Figure 107 below. The top computer is connected to the controller via its COM port. Both computers are connected to a TCP/IP network.

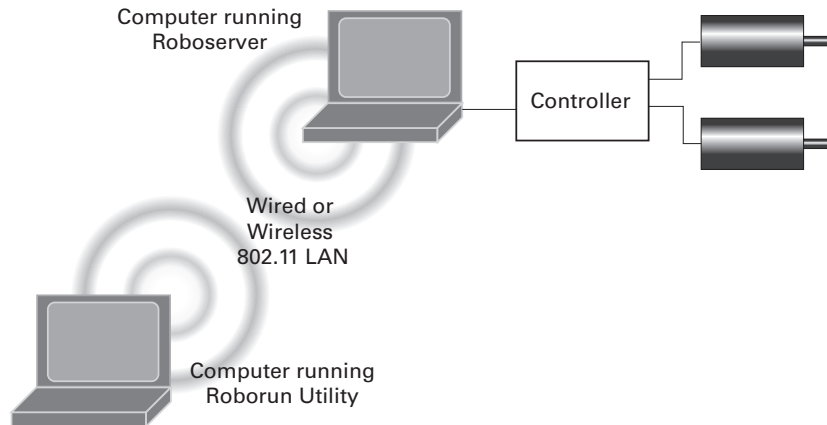


FIGURE 107. Operating the controller over a LAN

The computer connected to the controller must run a communication server program named Roboserver. This program is automatically installed in the Start menu when installing the Roborun utility. This program's function is to wait for and accept TCP/IP connection requests from the other computer and then continuously move data between the network and the COM port. When launched, the screen shown below appears.

The second computer runs the Roborun utility. To establish contact with the server program, click on the "Change COM/LAN Port" button and enter the IP address of the second computer. Communication should be established immediately.

When the two computers are connected, it will be possible to operate the motors and read the controller's operating parameters in the Roborun Run window.

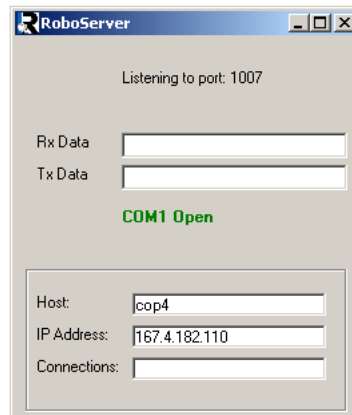


FIGURE 108. Roboserver screenshot when idle

Note that it is not possible to use this configuration to change the controller's parameters or update the controller's software.

## Updating the Controller's Software

The AX2550's operating software can be easily upgraded after it has left the factory. This feature makes it possible to add new features and enhance existing ones from time to time.

## Important Warning

**Updating the controller will cause all its parameters to reset to their default conditions. You should re-enter these parameters to the desired value prior to re-installing and using the controller.**

The upgrade procedure is quick, easy and error proof:

- 1- Connect the controller to the PC via the provided RS232 cable.
- 2- Apply a 12V to 40V power supply to the controller's Ground and Power Control input (thin black and yellow wires). Leave VMot disconnected.
- 3- Launch the Roborun utility if it is not already running. Then click on the "Update Controller Software" button.
- 4- If the controller is On, Roborun will find it and prompt the selection of the new software file. It may happen that the controller is not responding properly and you may be asked to reset it while connected.
- 5- Press the "Program" button to start programming. **Do not interrupt or cut the power to the controller while the new program is loading into Flash memory.**
- 6- After a verification, you will be notified that the operation was successful and you will see the new software revision and date as reported by the controller.

Notes:

The Updating utility will automatically detect whether the new software is intended for the main or encoder's MCU and program one or the other accordingly.

It is a good idea to load the controller's parameters into the PC and save them to disk prior to updating the software. After the new software is transferred to the controller, you can use the "Load Parameters from Disk" function and transfer them to the controller using the "Save to Controller" button.

## Updating the Encoder Software

The Encoder Module has its own dedicated MCU and software in Flash memory. It may be updated using the Roborun Utility in the same manner as for updating the controller's software (see "Updating the Controller's Software" on page 194). Then select the new software file to download. The file's content automatically identifies it as software for the Encoder MCU rather than the Controller's MCU.

## Important Warning

**Do not reinstall the same firmware version as the one already installed in the encoder module.**

---

## Creating Customized Object Files

It is possible to create versions of the controller's firmware with default settings that are different than those chosen by Roboteq. This capability can be used to improve system reliability in the unlikely, but not impossible, occurrence of a parameter loss in the controller's non-volatile memory. Should such an event occur, the controller would revert to the defaults required by the application.

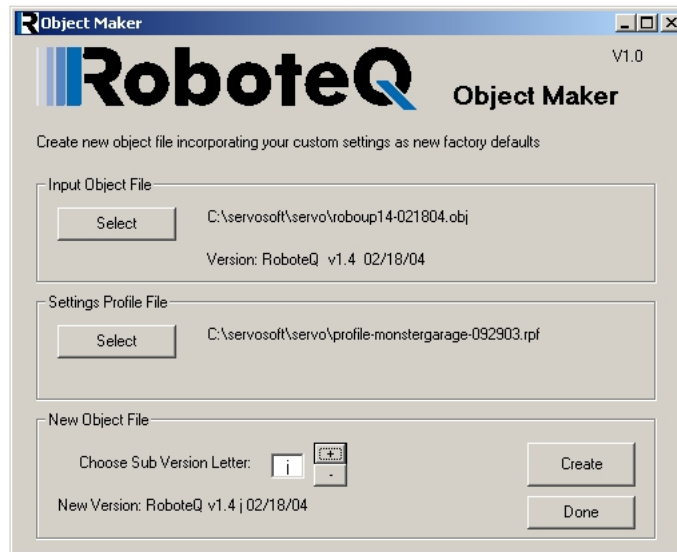


FIGURE 109. Objectmaker creates controller firmware with custom defaults

Creating a custom object file can easily be done using the Objectmaker utility. This short program is automatically installed in the Start menu when installing the Roborun utility.

- 1- Use the Roborun utility to create and save to disk a profile with all the desired parameter value.
- 2- Launch Objectmaker from the Start menu.
- 3- Select the latest official controller firmware issued by Roboteq.
- 4- Select the profile file that was created and saved earlier.
- 5- Select a revision letter. This letter will be added at the end of Roboteq's own version identity number.
- 6- Click on the Create button and save the new customized object file.
- 7- Click on the Done button to exit the program.
- 8- Install the new object file in the controller using the Roborun utility.

## SECTION 16

# Mechanical Specifications

This section details the mechanical characteristics of the AX2550 controller.

---

## Mechanical Dimensions

The AX2550 is built into a extruded aluminum case which also serves as a heat sink for its electronics. Figure 110 and Figure 111 show drawings of the controller and its mechanical dimensions.

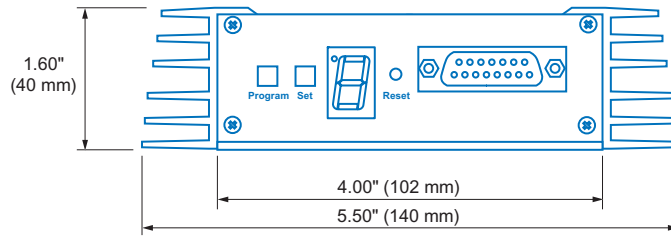


FIGURE 110. AX2550 front view and dimensions

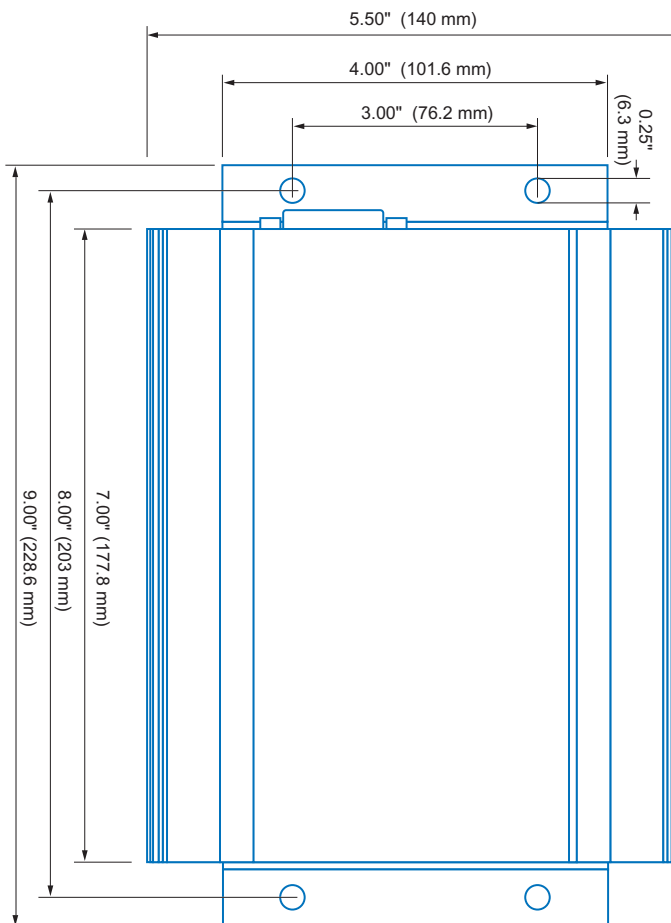


FIGURE 111. AX2550 top view and dimensions

## Mounting Considerations

Mounting brackets on each end are provided for durable assembly of the controller onto any chassis.

Optional shock mount absorbers may be added for use in the most brutal environment.

---

## Thermal Considerations

The AX2550 is built into a heat-sinking aluminum extrusion. The large fin area on the side of the extrusion ensures sufficient heat dissipation for operation without a fan in most applications.

For high current applications, it is possible that the controller may heat up faster and to a higher temperature than can be dissipated by the heatsink enclosure using natural convection alone.

In these applications, you should ensure that air flow exists to remove the heat from the heat sink. In the most extreme use, you should consider using an external fan to circulate air around the controller.

---

## Attaching the Controller Directly to a Chassis

The AX2550 can be attached to a metal chassis to improve heat dissipation. You may mount the controller so that its bottom area is in contact with large metallic parts of the robot's body. This will allow some of the controller's heat to dissipate through conduction to additional metal mass, therefore remaining cooler for longer periods of time. Apply thermal grease between the controller's bottom and the robot's chassis to ensure the best thermal conduction.

---

## Wire Dimensions

A total of eight heavy gauge wires, and two thin wires are present at the back of the controller. Their length and dimension is shown in Table 37 below.

TABLE 37. AX2550 wire identification and dimensions

Wire	Gauge	Outside Diameter	Color	Length
Motor1 Battery+	AWG8	0.26" (6.6mm)	Red	17" (43mm)
Motor1 Battery-	AWG8	0.26" (6.6mm)	Black	17" (43mm)
Motor2 Battery+	AWG8	0.26" (6.6mm)	Red	17" (43mm)
Motor2 Battery-	AWG8	0.26" (6.6mm)	Black	17" (43mm)
Motor1 Terminal+	AWG8	0.26" (6.6mm)	White	17" (43mm)
Motor1 Terminal-	AWG8	0.26" (6.6mm)	Green	17" (43mm)
Motor2 Terminal+	AWG8	0.26" (6.6mm)	White	17" (43mm)
Motor2 Terminal-	AWG8	0.26" (6.6mm)	Green	17" (43mm)

TABLE 37. AX2550 wire identification and dimensions

Wire	Gauge	Outside Diameter	Color	Length
Power Control or Backup Controller Battery	AWG22	0.07" (1.8mm)	Yellow	17" (43mm)
Controller Battery-	AWG22	0.07" (1.8mm)	Black	17" (43mm)

---

## Weight

Controller weight is 3.3 lbs (1500g)