

ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

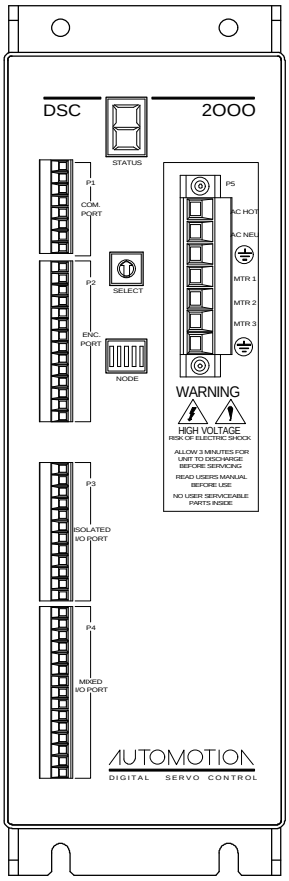
Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Visit our website - *Click HERE*

DSC2000 User's Manual

©1996 Automotion Incorporated
All Rights Reserved.



READ ALL SAFETY PRECAUTIONS FIRST.
(See Section 1.0)

WARNING

HIGH VOLTAGE
RISK OF ELECTRIC SHOCK

This manual covers general operation, software programming, and hardware installation of Automotion Digital Servo Control, model:

DSCxxxx-xxx-xxxx

This manual applies to serial numbers beginning after: **xxxxx0896**

This manual applies to Firmware Release: **REV 1.0**

Document Revision: 1.0a (On-line PDF format, 04/27/99)

Document Release Date: 8/20/96

Control # A10518

! Important: This publication is issued on condition that it is not copied, reprinted, or disclosed to a third party, either wholly or in part, without the prior written consent of Automotion Incorporated.

Automotion Incorporated assumes no liability or responsibility for the loss or damage, direct or indirect, arising from the use of this product. Automotion reserves the right to change this product's specifications without notice.

This document is based in information available at the time of its publication. While all efforts have been made to be accurate, the information contained in this guide does not purport to cover all details or variations in hardware or software. This guide may describe features which are not present in all hardware and software systems.

Automotion Standard Terms and Conditions apply.

IBM and PC are registered trademarks of International Business Machines Corporation. Microsoft and Windows are registered trademarks of Microsoft Corporation.

Mail: Automotion Incorporated
P.O. Box 7746
Ann Arbor, Michigan USA 48107

Sales/Service: Automotion Incorporated
3808 Plaza Drive
Ann Arbor, Michigan USA 48108

Phone: (734) 662 - 7771

Fax: (734) 662 - 3707

E-mail: sales@automotioninc.com

WEB Site: www.automotioninc.com

©1996 Automotion Incorporated, All Rights Reserved. Printed in U.S.A.

Section 1.0 Introduction *Page Number*

Safety Precautions 1-1
Warranty 1-3
General Product Specifications 1-4
How this Manual is Organized 1-6

Section 2.0 Getting Started

Theory of Operation 2-1
Product Overview 2-2
Minimum Setup 2-5
Connecting to the DSC 2-9
First Time Operation 2-11
Creating Motion 2-15
DSC Trouble Shooting 2-18

Section 3.0 DSC Command Interpreter

Memory Organization 3-1
Command Structure and Syntax 3-2
Command Addressing Modes 3-3
Command Types 3-4
Command Strings 3-5
Macro Commands 3-6
Stand Alone Operation; Macro 0 3-7
Point to Point Serial Communications 3-8
Multidrop Serial Communications 3-9

Section 4.0 DSC Servo Control

General Control Architecture 4-1
Current Loop Operation 4-2
Position Loop Operation 4-3
Post Position Loop Operators 4-4
Position Mode 4-5
Velocity Mode 4-6
Gain Mode 4-7
Torque Mode 4-8
Slave Mode 4-9
Contour Mode 4-10
Suggestions for Servo Loop Tuning 4-11

Section 5.0 DSC Command Summary.

Group Listing 5-1
Parameter Setup Commands 5-5
Motion Commands 5-17
Reporting Commands 5-26
Macro Commands 5-39
Channel commands 5-43
Sequence Commands 5-46

Learn Mode Commands 5-57
Register Commands 5-58
Miscellaneous Commands 5-73

Section 6.0 DSC Hardware Reference

Package Outline 6-1
General Installation 6-2
Communication Port, P1 6-3
Encoder Port, P2 6-5
Isolated I/O Port, P3 6-6
Mixed I/O Port, P4 6-8
Power Port, P5 6-10
External Shunt Installation (Optional) 6-11

Appendices

Alphabetical Command Reference ..... A
Firmware Default Settings, Servo Table ..... B
Firmware Revision History C
Fault Codes, Operation and Recovery ..... D
DSC to SP Motor Switch Settings ..... E
DSC Commutation Sequence Tables ..... F
Suggested method for current loop tuning ..... G

Section 1.0

Introduction



SAFETY PRECAUTIONS

READ THIS SECTION BEFORE USING THE DSC2000

To operate the DSC2000 control successfully, these minimum safety precautions MUST be followed to insure proper performance without injury to the operator and damage to motor or control. FAILURE TO OBSERVE THESE SAFETY PRECAUTIONS COULD RESULT IN SERIOUS BODILY INJURY, INCLUDING DEATH IN EXTREME CASES.

1. DO NOT touch any of the power connector pins of port P5 when power is applied. The voltages at these connector pins are dangerous and can produce an electric shock. Bare wires from adjacent connector pins must never be allowed to touch one another. Pins marked with ground symbol at Port P5 must be connected to an external earth ground. Follow wiring procedures carefully. Know and understand which ports are NOT electrically (galvanically) isolated from the AC/DC power.

2. Always operate the control within the prescribed voltage limits.

3. Each model has dangerous voltages on the circuit boards and stores a high voltage charge after being disconnected. DO NOT REMOVE THE COVER IF ONE IS IN PLACE. DO NOT ATTEMPT TO SERVICE THIS PRODUCT IF A PROBLEM OCCURS.

4. Do not parallel multiple motors off of the same control.

5. Under no circumstances must a phase output from the control be connected to anything other than a passive inductive/resistive motor load. See manual for minimum inductance requirements.

6. Excessive speed and current can destroy some DC brushless motors and possibly injure the user. Check the motor manufacturer's specifications to ensure the maximum current and voltage output for your control model does not exceed motor limitations.

7. Methods must be employed by user to limit both the top speed and travel motion of the motor and its load. The DSC2000 product does not limit motor speed or travel automatically. Whenever the DSC2000 is disabled for any reason, the motor is placed into a free/spinning coast mode.

8. Do not remove the connectors on ports P1, P2, P3, P4 or P5 from the control while the motor is operating.

9. Read Automotion's Life Support Policy for application limitations.

10. To avoid a shock hazard always wait at least 15 minutes after disconnecting power into port P5 before physically handling or touching any internal circuit boards.

11. Follow precautionary guidelines in this manual with regard to proper installation of an external shunt resistor. See Section 6.0.



SAFETY PRECAUTIONS (continued)

12. If the control does not seem to function correctly first consult section titled, "DSC Trouble Shooting". If this fails to solve the problem, call Automotion.

13. DO NOT LOCATE ANY DSC2000 CONTROL IN A POSITION WHERE IT WOULD HAVE CONTACT WITH LIQUIDS, ELECTRICALLY CONDUCTIVE MATERIALS, WATER CONDENSATION, CORROSIVE CHEMICALS OR WHERE FOREIGN MATERIALS WOULD BE ALLOWED TO FALL INTO OR COLLECT WITHIN THE DSC2000.

14. DO NOT MOUNT THE DSC2000 CONTROL DIRECTLY UPON OR NEAR FLAMMABLE MATERIALS.

15. DO NOT OPERATE THE DSC2000 CONTROL IN AN EXPLOSIVE ATMOSPHERE OR IN THE VICINITY OF EXPLOSIVE MATERIALS. KEEP THE INSTALLATION VENTILATED SO THAT CLEAN FRESH AIR CAN MOVE FREELY THROUGH THE DSC2000 CONTROL ENCLOSURE.

16. Do not plug port connector P5 into the control while live power is applied. This can cause electrical arcing which can damage the connector pins. Automotion recommends using a disconnect switch ahead of port P5 to absorb current surge if the power source must be interrupted often.

LIFE SUPPORT POLICY:

AUTOMOTION'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF AUTOMOTION INCORPORATED.

AS DESCRIBED HEREIN:

Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the User's Manual and in the labeling, can be reasonable expected to result in a significant injury to the user.

A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

Product Warranty

Automotion Incorporated hereafter referred to as the "Company", warrants its product for a period of 2 years from date of shipment to be free of defects caused by faulty material or inferior workmanship. The liability of the Company under this warranty extends to the replacement, repair, or issuance of credit, at the Company's option, for any of its products which are returned by the initial Purchaser during such period, provided:

The Company is promptly notified in writing upon discovery of such defects by Purchaser.

The defective product(s) is returned to the Company after the Company has issued a Return Authorization Number, with all return transportation charges prepaid by the Purchaser.

The Company's examination of such product(s) shall disclose to its satisfaction that such defects exist and have not been caused by improper installation, neglect, repair, alteration, misapplication, or accident caused by the Purchaser.

Under no circumstances shall the Company be liable for the collateral or consequential damages of any nature incurred through the use of our products

Automotion's Standard Terms and Conditions also apply.

Unpacking and Repacking

When the DSC2000 arrives, inspect the shipping box and the unit carefully, and save ALL packing materials. Contact the carrier promptly if damage is discovered.

The DSC2000 is carefully packaged from Automotion in an antistatic bag. When unsealing this bag, inspect the contents carefully. There should be no loose or damaged parts inside. Compare the packing slip against all items included in the box. Reported any shortages or other inspection problems to Automotion immediately.

Never attempt to operate or power-up the DSC2000 if there is any visible external damage, or if it sounds as though there are loose materials inside the chassis. While unpacking, if any loose or damaged parts are discovered, notify Automotion within two working days.

Automotion recommends that all packing materials be saved in case the DSC2000 ever needs to be shipped again. Always place the DSC2000 in the same antistatic bag used in the original shipment. Abundant filler material should always be placed around the DSC2000 bag so that it cannot shift inside the box. Exercise care when placing packing material around port connectors to prevent stress damage due to shock and vibration.

Always insure your shipment for the proper replacement value of its contents. Automotion will not assume responsibility for any returned goods which have been damaged outside of our factory due to improper packaging or handling. All goods shipped to Automotion must have prior return authorization numbers and be shipped freight prepaid.

The following is a brief overview of DSC product specifications. Reference Section 6 in this manual for a detailed electrical specification of DSC Port connections.

CONTROL CARD

32 MHz 16 bit microcontroller
On board Watchdog circuit
128 Kbyte Flash
32 Kbyte Ram
7 segment Status display
10 position Select switch input
5 bit input Node switch

PORT DESCRIPTION

P1: RS232/RS485 serial com. port.
P2: Encoder Feedback
P3: Isolated Input(8)/Output(1)
P4: Mixed Input(2)/Output(4)
P5: Power Input/Output

SERVO CONTROL

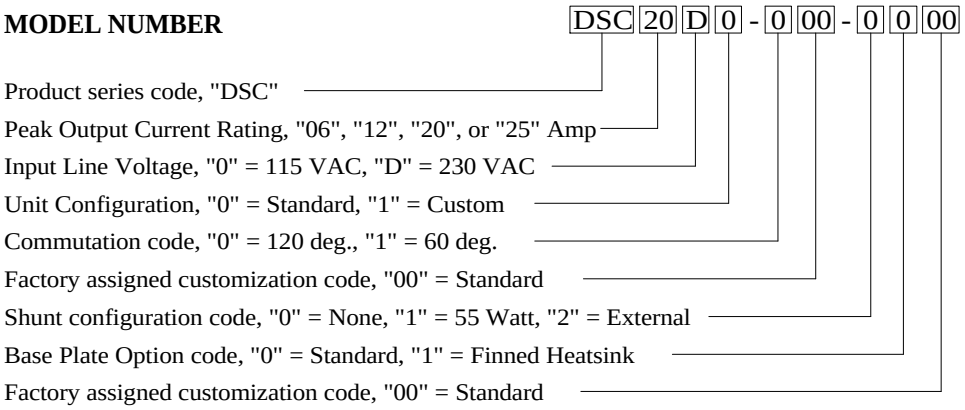
Current Resolution	11 bits
Current Loop Bandwidth	Up to 2 KHz
Encoder Feedback	1 MHz
Acceleration	32 bits, 16 bit fractional
Velocity Resolution	32 bits, 16 bit fractional
Position Resolution	32 bits integer
PID Filter Update Rate	2 KHz
Trajectory Modes	(6) Position, Velocity, Gain, Torque, Slave, Contour
Feedforward	Acceleration, Deceleration and Velocity
Torque Operators.	Output deadband, Output offset, Torque limiting

COMMAND INTERPRETER

Interpretive, 150+ Commands	Parameter Setup, Motion, Reporting, Macro, Channel(Input/Output), Sequence, Learn, Register.
Macro Storage	256 User defined command strings.
Registers	256 General Purpose 32 bit registers.
Servo Table	User accessible, real time Servo data.
Arithmetic Operation	Add, Subtract, Multiply, Divide (see Register type).
Logical Operation	AND, OR, EOR, NOT (see Register type).
Branching Operation	DO if on, DO if off, IF on, IF off., Macro Jump, Macro Call. (see Sequence and Macro types).
Process Operation	Supporting 3 Analog inputs, Auxiliary Position or Count input.
Interrupt Driven Functions	Motion sequencing from external input, Output sequencing from servo position (see Sequence type)
Automatic Fault Monitoring	Logic brownout, Processor watchdog, Reset active, Following Error, Motor Temp, Amplifier, Command error(12), Travel Limit.
Real Time Data Recording	2047 Points: Desired Position, Actual Position and Torque. Variable recording time base, up to 2 KHz.
Operational Modes	Immediate (serial interface), Standalone (user programmed).
Communication Modes	Point to Point, Multidrop.
Communication Format	ASCII decimal, or hexadecimal.

"DSC2000" or "DSC" are used throughout this manual interchangeable and refer to all model numbers within the product series. Reference the Model Number description given below for specifications unique to any given unit.

MODEL NUMBER



MODEL RATINGS

MODEL	**DSC0600**	**DSC1200**	**DSC2500**	**DSC06D0**	**DSC12D0**	**DSC20D0**
Input Voltage	90-132 VAC	90-132 VAC	90-132 VAC	180-264 VAC	180-264 VAC	180-264 VAC
Input Line	1 Phase, 50/60 Hz	1 Phase, 50/60	1 Phase, 50/60	1 Phase, 50/60	1 Phase, 50/60	1 Phase, 50/60
Bus Voltage	160 VDC	160 VDC	160 VDC	320 VDC	320 VDC	320 VDC
Peak Output	6 Amp	12 Amp	25 Amp	6 Amp	12 Amp	20 Amp
Cont Output	4 Amp	8 Amp	15 Amp	4 Amp	8 Amp	12 Amp
Regeneration Capacity	55 Watt Option: 40 cont./800 peak watts, thermally protected External option: up to 400 cont/ 3000 peak.					
Bridge	Full Bridge, Isolated Gate Bipolar Transistor (IGBT)					
Motor Ind.	250 uH minimum			500 uH minimum		
Modulation	18 KHz PWM (standard) to 40 KHz (optional)					
Commutation	6 Step, Trapezoidal, 120 degree (standard), 60 degree (optional)					
Protection	Phase to Phase, Phase to Ground, Under/Over Voltage, Regeneration					
Isolation	1500 VAC, 1 min.: VAC Line (P5) to Frame Ground 1500 VAC, 1 min. VAC Line (P5) to Logic (P1, P2, P3 & P4)					
Temperature	0 to +50 C (operating), -20 to +85 C (storage)					
Weight	9.4 lb. typical (4.2 Kg)					

First Time Users

Section 2.0, Getting Started, is a good place to begin. This section provides a product overview to familiarize the user with DSC features. This section also describes a minimum bench setup for evaluation and initial operation. The balance of Section 2.0 is written as a tutorial which steps the user through basic hookup and DSC operation.

Using this Manual as a Reference

Section 3.0 and 4.0:	A formal reference to DSC firmware operation . Section 3.0 describes foreground operation, command interpretation. Section 4.0 describes background operation, servo control.
Section 5.0:	All DSC commands are individually described in this section. Commands are organized first by group (Parameter Setup, Motion, ...etc.) and then alphabetically. For an alphabetical listing of commands independent of group see Appendix A.
Section 6.0:	Reference Section 6.0 for information on DSC electrical specifications and installation.

Conventions used in this Manual

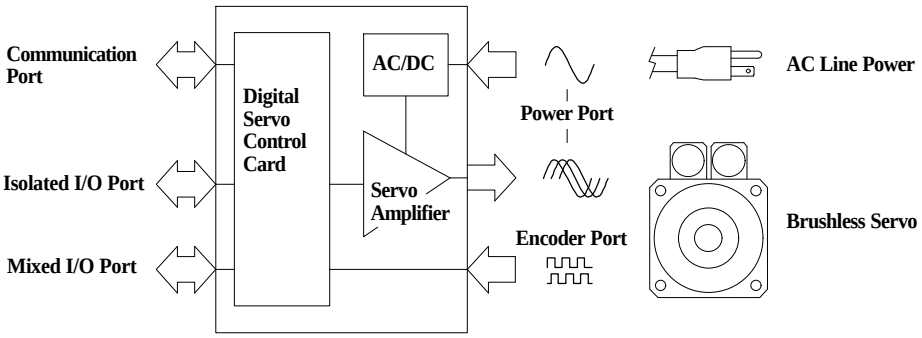
! Important:	"**! Important:**" appears in the left hand margin on many pages in this manual. Read and understand this information. Improper use of the DSC2000 can result in personal injury and, or damage to equipment.
>	The ">"character appears in many DSC program and command examples. This is the standard DSC serial port prompt character and is not typed or entered by the user.
<*Enter*> or <*CR*>	These are used interchangeably throughout this manual. They indicate that the user should press the enter key, or transmit an ASCII carriage return character.
+	The "+" character appears in many programming examples. When a "+" is appended to a DSC command, it should not be typed or entered by the user. Rather, this character is used to indicate that subsequent lines are to be appended.
;	The "**;**" character appears in many programming examples. This character, and those that follow on the same line, are considered comments and are not entered or transmitted to the DSC.

Section 2.0

Getting Started

The DSC2000 is an integrated, programmable, single axis servo control. Internally, the DSC2000 is constructed from several fundamental building blocks. These include a servo control card, a torque mode brushless servo amplifier, and a single phase AC to DC high voltage supply.

DSC2000



The AC to DC supply rectifies and filters AC line voltage to produce an internal DC power source. The servo amplifier draws upon this source to control a three phase, permanent magnet, brushless motor. The servo amplifier provides commutation, modulation and closed loop control of current. By controlling motor current, the amplifier effectively controls motor torque. This "torque" control loop serves as the inside loop of a dual loop control system.

The servo control card forms an outer, or second, control loop around the amplifier. The control card senses motor position at the encoder port, and generates a torque command for the amplifier. This torque command holds the motor at a "desired" position. A PID (Proportional, Integral, Derivative) algorithm is incorporated into the DSC to accomplish this task.

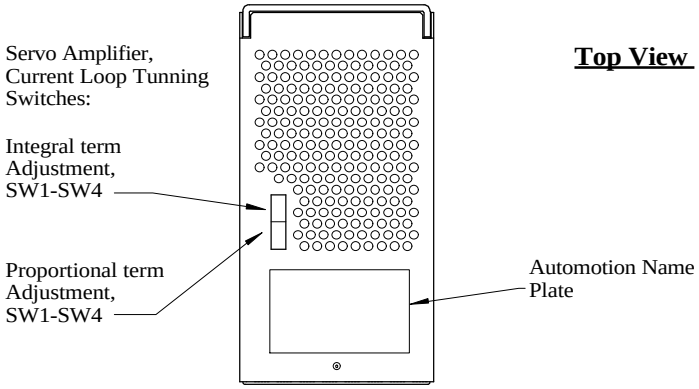
Also incorporated into the DSC, is a motion path, or "trajectory generator" function. The trajectory generator function operates on desired position as a variable. By incrementally changing desired position over time, the trajectory generator drives the motor from one position to another.

Holding and driving motor position are background functions of the digital control card. This card divides processing time into foreground and background operation, working a little on both, 2000 times per second. In the foreground, the control card interprets user commands. These commands allow the DSC2000 to interact with input/output devices, specify motion activity, and report on operation.

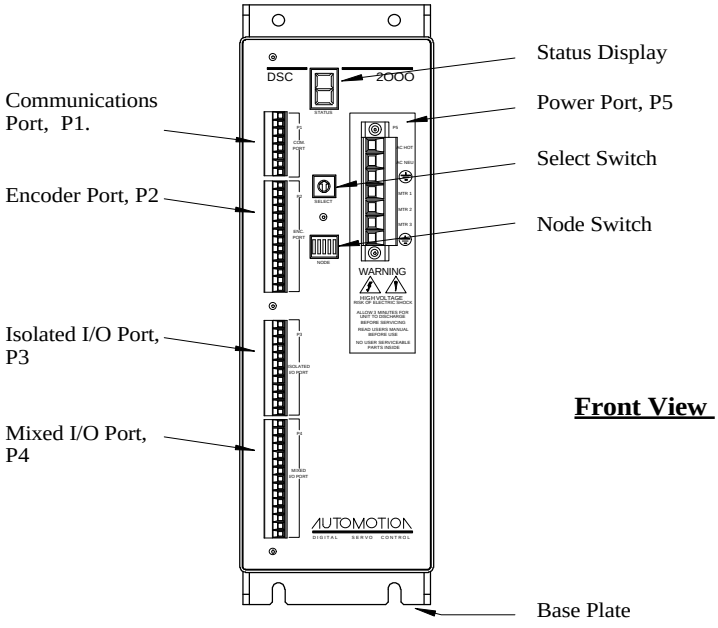
DSC2000 commands may be issued over the serial port, where they are executed in an immediate fashion, or they may be stored in resident, nonvolatile memory, where they are executed in a standalone fashion. More than 150 unique software commands, and six unique trajectory generation modes, are supported by the DSC2000.

Section 3 of this manual, DSC Command Interpreter, covers foreground operation of the control card. Section 4, DSC Servo Control, covers background operation. The remainder of this section serves as an orientation to the fundamental product features and concepts within the DSC2000.

The DSC2000 is designed to panel mount inside an industrial control enclosure. Control connectors and associated signals are grouped by function. These connectors are screw terminal type and are suitable for direct termination of internal control panel wiring. Reference Top and Front View drawings for the following summary:



Top View



Front View

Current Loop, Integral Term Adjustment, SW1-SW4

SW1 through SW4 in this switch bank determine the amount of integral gain used by the servo amplifier for controlling motor current. Proper setting of these switches depends on the specific DSC2000 model number and servo motor being used. If using an Automotion SP series servo motor, reference appendix E for a predetermined list of switch settings. If mating the DSC2000 to a compatible servo

motor, reference section 4, Current Loop Operation, or consult Automotion for recommended switch settings. Once these settings are determined, they remain constant for a given DSC2000 and servo motor package.

Current Loop, Proportional Term Adjustment, SW1-SW4

SW1 though SW4 in this switch bank determine the amount of proportional gain used by the servo amplifier for controlling motor current. As with the integral term adjustment, proper setting of these switches depends on the specific DSC model number and servo motor being used. If using an Automotion SP series servo motor, reference appendix E for a predetermined list of switch settings. If mating the DSC2000 to a compatible servo motor, reference section 4, Current Loop Operation, or consult Automotion for recommended switch settings. Once these settings are determined, they remain constant for a given DSC2000 and servo motor package.

Automotion Name Plate

The DSC2000 name plate is located on the top side of the unit. This name plate lists a specific model number, serial number, and revision level. The numbers and letters within the model number, DSCxxxx-xxx-xxxx, specify the particular product options that exist on a given unit. See General Specifications for decoding. Please reference name plate information when applying the DSC, or calling Automotion for service.

Communications Port, P1

Connector P1 serves as the serial communications interface to the DSC. Three unique communication modes are supported; RS232 point to point, RS485 point to point, or RS485 multidrop. Terminal P1-8 functions as a RS485 driver select. The setting of Node switch number 5 configures the DSC for either point to point, or multidrop operation. Reference section 6 for an electrical specification of this port. Reference section 3 for software operation.

Encoder Port, P2

Connector P2 serves as the servo motor feedback interface to the DSC. Signals supported at this interface include three differential encoder signals, three motor commutation signals, and one motor temperature signal. Reference section 6 for an electrical specification of this port.

Isolated I/O Port, P3

Connector P3 serves as an isolated input/output port for the DSC. Signals supported at this interface include Enable/Reset, Limit+, Limit-, 5 user programmable inputs, and a fault relay output. Reference section 6 for a hardware specification of this port.

Mixed I/O Port, P4

Connector P4 serves as an mixed input/output port for the DSC. Signals supported at this interface include one deferential analog input, one auxiliary feedback input (quadrature encoder, or step and direction), and 4 programmable outputs.

Logic power is also supplied by the user at this interface. Reference section 6 for a hardware specification of this port.

Power Port, P5

Connector P5 serves as both an AC line input, and motor phase output, power interface. Reference DSC name plate ratings and section 6 for additional specifications.

Status Display

The status display enunciates letter codes when the DSC detects a system fault condition. When no system fault condition is present, the user can program this display with application specific messages . Reference appendix D for a summary and discussion of DSC system faults. See section 5, UD command, for information on writing to the status display.

Select Switch

This 10 position rotary switch is a general purpose input device to the DSC. This switch can be read over the serial interface, or incorporated into application program, to facilitate conditional program execution or selection. See section 5, GS command, for information on reading the select switch.

Node Switch

This 5 position binary switch is used to configure the DSC serial interface for multidrop operation. Switch 5 enables multidrop mode, and switches 1 through 4 set a binary network address. Reference section 3 for additional information on multidrop operation.

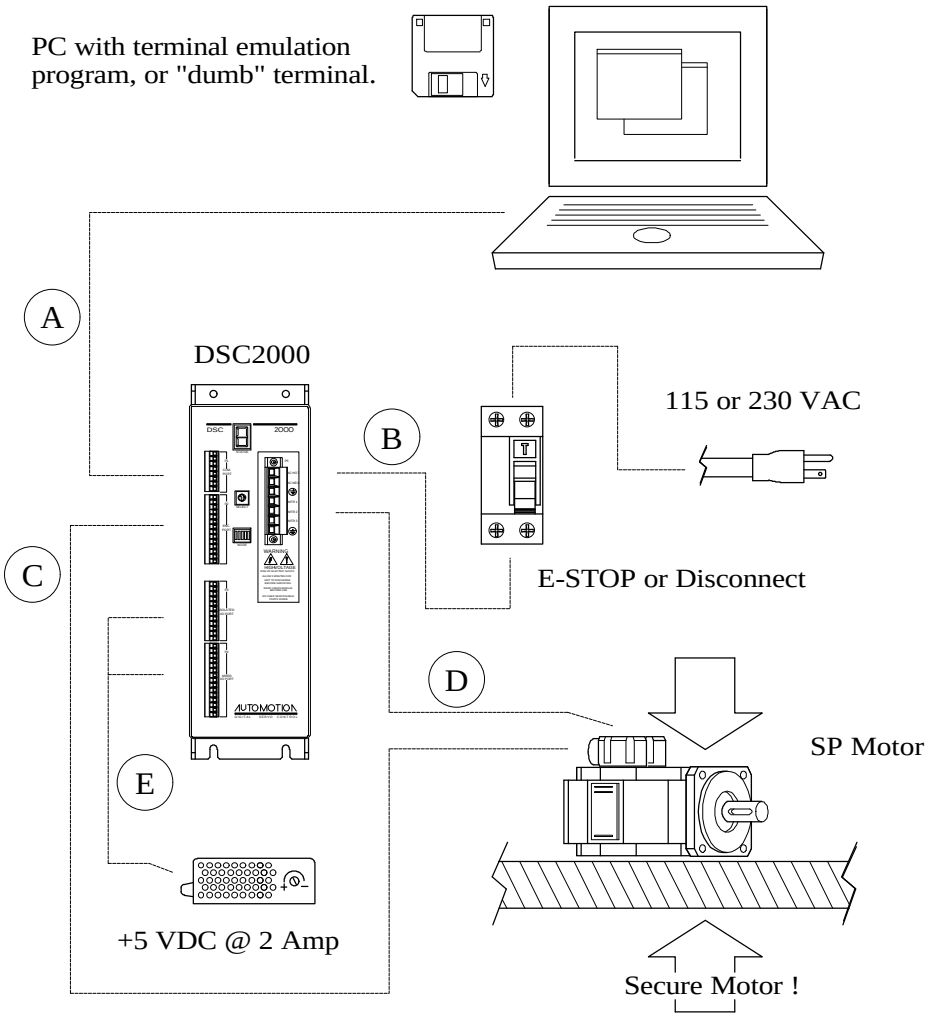
When the DSC is configured for point to point serial communication, switches 1 through 4 can be read over the serial interface, or incorporated into application program, to facilitate conditional program execution or selection. See section 5, GN command, for information on reading the select switch.

Base Plate

Most heat producing devices within the DSC are mounted to the base plate. See section 6, DSC Installation for proper mounting and application of the DSC2000.

! Important When working with the DSC2000 for the first time, Automotion recommends starting with the following minimum setup. The servo motor should be secured to a bench top, or suitable surface, with no load connected to the shaft. The bench should be free from wire or any material that could be caught on the motor shaft. Automotion also recommends locating the DSC2000 where it can be observed, but where it is safely mounted, and presents no electrical shock or fire hazard.

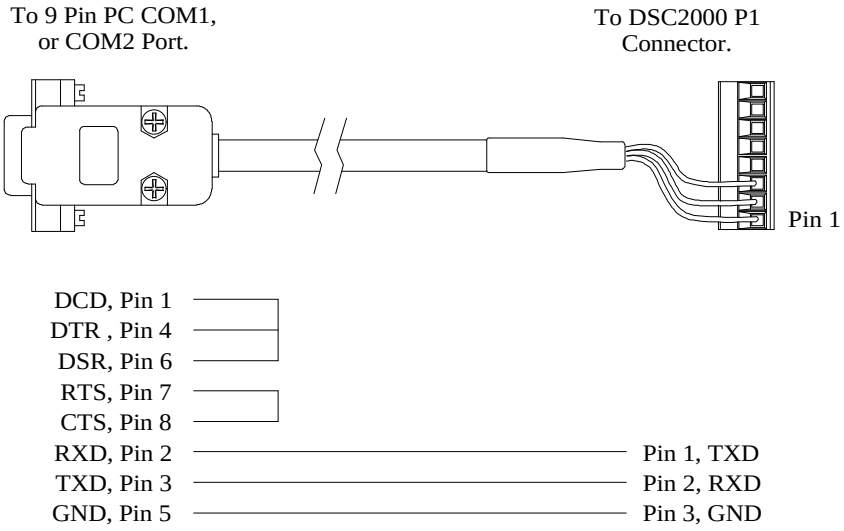
The following items will need to be located. Letters "A, B, C..." denote cable assemblies.:



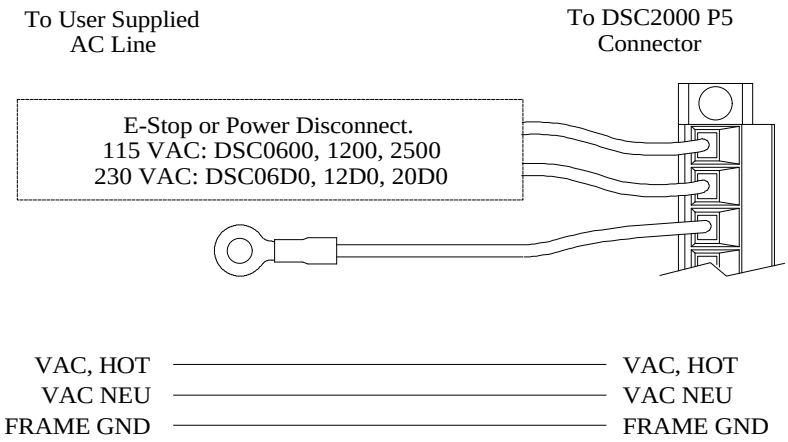
Automotion recommends using an "E-Stop" or power disconnect wired in series with AC line power to the DSC. Servo operation can be aborted by removing AC power from the DSC. The DSC control card and motor feedback are independently power from an external 5 VDC supply. This feature allows the user to interrogate DSC status and motor position when AC line power is removed.

! Important Issuing software commands when learning about DSC operation, or debugging an application program, can cause unexpected motion. Place the E-Stop or power disconnect within hands reach.

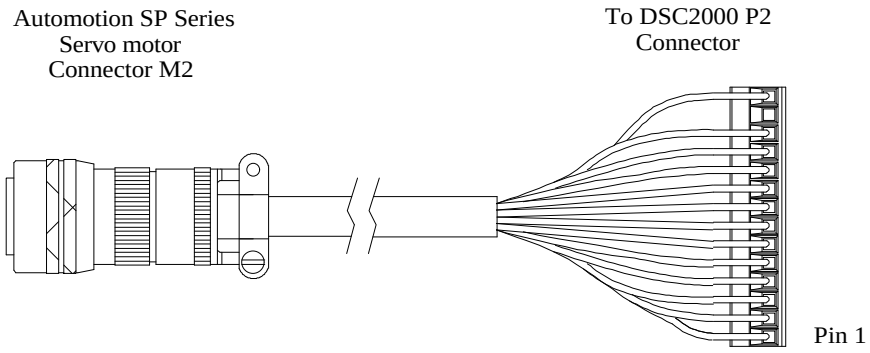
Cable A: **RS232 Serial Communication Cable**



Cable B: **Power Entry Harness**

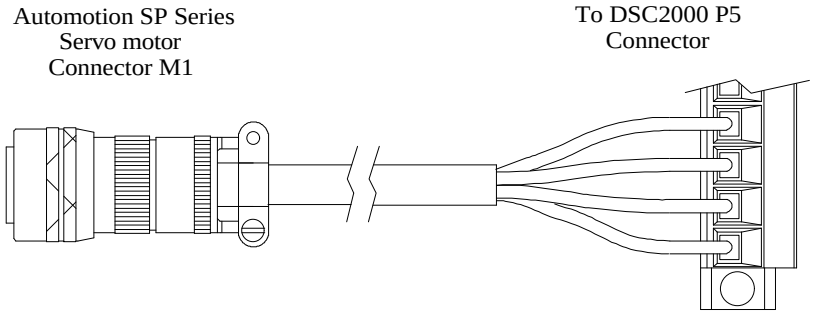


Cable C: **Motor Feedback Cable**



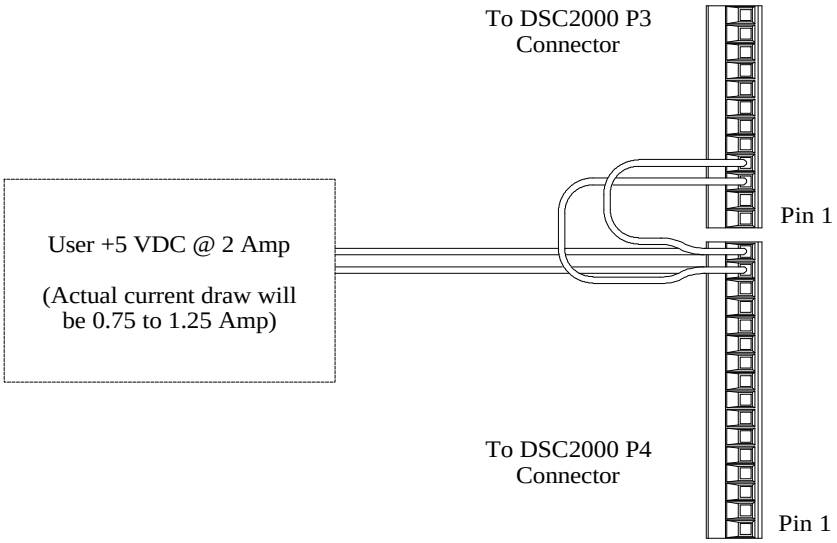
M2, Pin B	Pin 1, +5 VDC
M2, Pin A	Pin 2, Enc. Com.
M2, Pin D	Pin 3, Enc. A
M2, Pin C	Pin 4, Enc, !A
M2, Pin F	Pin 5, Enc. B
M2, Pin E	Pin 6, Enc. !B
M2, Pin H	Pin 7, Enc. Z
M2, Pin G	Pin 8, Enc. !Z
	Pin 9, Shield
M2, Pin K	Pin 10, S1
M2, Pin M	Pin 11, S2
M2, Pin P	Pin 12, S3

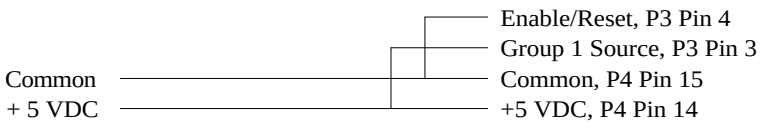
Cable D: **Motor Power Cable**



M1, Pin 2	MTR 1
M1, Pin 3	MTR 2
M1, Pin 1	MTR 3
FRAME GND	FRAME GND

Cable E: **Minimum I/O Harness**





The Following procedure describes how to connect the DSC2000 to an RS232 terminal device, and verify operation:

Step 1: Configure the DSC for RS232 operation.

Connect the "RS232 Serial Communication Cable" to COM1 or COM2 of a PC, or compatible device. Connect the "P1" end of this cable to the DSC. Inspect wiring for loose or shorted connections.

Check DSC Node switch settings. This device is located on the DSC face plate between connectors P2 and P5. Place Node switch number 5 in the "off" or down position.

Step 2: Connect +5 VDC logic power to the DSC.

Before connecting the "Minimum I/O Harness", verify that the logic supply voltage is adjusted to between +5.0 VDC and +5.2 VDC. Turn this supply off, and then connect P3 and P4 to the DSC. Inspect wiring for loose or shorted connections.

Turn the +5 VDC supply on. All segments of the DSC Status display will momentarily light and then just the decimal point will turn off. IF a "b." is displayed, then the 5 VDC supply drooped during power up. Readjusted to between +5.0 VDC and +5.2 VDC, and cycle logic supply power switch again (i.e. on/off/on).

If an "r." is displayed, check connections at P3 and P4 and cycle logic supply power again. Display codes "b." and "r." represent "logic brownout" and "reset input active".

Step 3: Configure the PC COM port or terminal device.

The default DSC2000 communication settings are follows.:

Baud Rate	Parity	Data Bits	Stop Bits	Flow Control
9600	none	8	1	none

If using a PC and Windows, double click on the terminal icon which is typically located in the accessories window. Communication settings are in a dialog box under the "Settings" menu. Also select COM1 or COM2.

By default, the DSC operates as a Full Duplex device which echoes received characters, including carriage returns, back to the PC or terminal. However, the DSC does not append line feed characters to carriage returns. If using the Window's terminal emulator, mark the "CR to CR/LF, INBOUND" box under the "Settings, Terminal Preferences..." menu, to append line feed characters.

Step 4: Verify the communications link.

When logic power is applied to the DSC, the DSC control card initializes local parameters and then attempts to run a user installed program. If no program is found, the control card will wait at the serial port for user commands.

With steps 1 through 3 complete, press the PC or terminal "Enter" key. The DSC responds by echoing the carriage return character, and then transmitting a prompt character, ">".

If no prompt is received, cycle +5 VDC logic power, wait for the Status display decimal point to blink, and press enter on the terminal again. If still no prompt, the

DSC may be executing a user program. Press the "Esc" key to abort operation of any program that may be executing.

When a prompt character, ">", is returned, type "VE" and press Enter. The revision level for installed DSC2000 firmware will be displayed. At the next DSC prompt, type "UD0" (i.e. UD<*zero*>) and press Enter. This DSC command clears the Status display.

Example:

```
<Enter>
><Enter>
>VE<Enter>
DSC2000 Digital Servo Controller   VER. AM1   REV 1.0
COPYRIGHT (c) Automotion Inc. 1995
ALL RIGHTS RESERVED
>UD0<Enter>
>
```

Step 5: Check for installed programs.

The DSC may be initialized with one or more factory installed programs. DSC programs are referred to as "Macros". Type the "tell macro, all" command, "TM-1", at the DSC prompt. A list of installed macros will be returned. Macros are numbered 0 to 255 and begin with the macro define command, "MD<*number*>, <*command*>, <*command*>,..."

Example:

```
>TM-1<Enter>
MD255,PM,SG500,SD4000,SI0,...      {macro 255 is installed}
>
                              {-or-}

>TM-1<Enter>
>                                  {no macros installed}
```

If one or more macros are installed, copy them from the terminal log file and paste them to a text file for future reference. Macros are stored in Flash memory. Unless Flash memory is reset by the RM command, macros can only be programmed once. Be aware of any used macro numbers for future reference.

The following procedure describes how to connect a servo motor to the DSC2000, verify operation, and initially tune the position PID loop. The DSC must be connected and communicating with a PC or terminal device. If not, see previous section before proceeding:

Step 1: Connect motor feedback cable.

Momentarily remove +5 VDC logic power from the DSC. Connect the "motor feedback cable" as shown in minimum setup between SP motor connector M2 and DSC connector P2. Apply +5 VDC logic power to the DSC. The motor feedback device, or encoder, presents an additional load to this supply. Verify that this supply remains regulated with the motor feedback cable connected.

Step 2: Check encoder feedback.

Verify the serial communication link to the DSC by pressing Enter a few times. At the DSC prompt, type "DH" and then press Enter. This command tells the DSC to define the present motor position as home, and equal to zero.

At the next prompt, type "TP" and the press enter. This command tells the DSC to report the present position of the servo in units of quadrature encoder counts. If the shaft has not moved since entering DH, the reported value will be zero.

! Important: If connected, remove AC line and motor phase power from the DSC by unplugging connector P5. Rotate the motor shaft clockwise about one revolution. Again type "TP" and press Enter. If the encoder is functional and properly phased, the DSC will report approximately 4000, or 8000 encoder counts, depending on the SP motor connected.

Note: If the reported value is negative, then standard SP encoder phasing is inverted. Inverted encoder phasing will cause positive feedback in the DSC servo loop, which will cause the servo to "run away" when enabled. Inspect connections at P2 for signal pairs that may be swapped (i.e. A & !A, B & !B, or A+!A swapped with B+!B).

Now rotate the motor shaft counter clockwise about two revolutions. Again type "TP" and press Enter. The DSC will report approximately -4000, or -8000 encoder counts, depending on the SP motor connected If using a compatible motor, the reported values will be 4 times the encoder line count specification (i.e. 4 x Enc. PPR).

Example:

>DH<*Enter*>	{*define as home and equal to 0*}
>TP<*Enter*>	{*tell position*}
0000000000	
>TP<*Enter*>	{*CW 1 revolution, tell position*}
0000003959	
>TP<*Enter*>	{*CCW 2 revolutions, tell position*}
-0000004122	
>	

Step 3: Set current loop tuning switches

Configure current loop tuning switches, integral term SW1-SW4 and proportional term, SW1-SW4. If using an Automotion SP series motor reference appendix E for a predetermined list of switch settings.

If connecting to a compatible motor, reference section 4, Current Loop Operation, or consult Automotion for recommended settings.

Step 4: Wire P5 power connector.

Connect the "motor power cable" (see minimum setup) between SP motor connector M1, and DSC power connector P5. Wire the "power entry harness", using a power disconnect or E-Stop device, to P5. Locate the power disconnect or E-stop within hands reach. Be prepared to use this device if undesired motion occurs. With P5 properly wired, apply AC line power to the DSC.

Step 5: Check motor phasing.

Verify the serial communication link to the DSC by pressing Enter a few times. At the DSC prompt, type "SQ256" and press Enter. This command limits maximum drive current, hence maximum motor torque, to about 25% of rated peak.

Enable the servo motor by typing "MN". Observe the motor shaft when pressing Enter. If no motion occurs, type "MR500", and again observe the motor shaft when pressing enter. MR500 tells the DSC to move the motor shaft 500 encoder counts relative to present shaft position (i.e. 45 mechanical degrees for a 1000 PPR encoder, or 22.5 mechanical degrees for a 2000 PPR encoder).

If the motor momentarily runs away, then coasts to a stop, and an "F." is displayed for DSC status, encoder phasing is incorrect. See DSC trouble shooting and repeat step 4.

If the servo appeared to move, enter the "MR500" command several more times. Motion will occur each time this command is entered. If motion only occurred the first time MN or MR500 was entered, then motor power phasing may be incorrect. See DSC trouble shooting and repeat step 4.

Step 6: Tune servo position loop.

In step 5, motion generated by entering "MR500" may appear to be loose or poorly controlled. Stiff, tightly controlled motion can be achieved by "tuning" the servo loop PID parameters. The following is a minimal method for achieving tuned control. See section 4, DSC Servo Control, for a more detailed approach.

As discussed in Theory of Operation, the DSC drives servo position using a trajectory generator function, and controls servo position using a PID algorithm. One method for evaluating PID tuning is to "drive" the servo with a step function, and observe the controlled, or "step response". By repeating this test many times, the user can determine the PID parameter values required for tight control.

A step function is created by placing the DSC into "gain mode". This is accomplished by typing "GM" and pressing Enter. In gain mode, the MR***n*** command tells the DSC to create a move (or "trajectory") by immediately updating "desired position" to "present motor position + ***n***". In this way, servo position can be "stepped" by ***n*** encoder counts, and servo response can be observed, as motion from start to end is controlled by the PID algorithm alone.

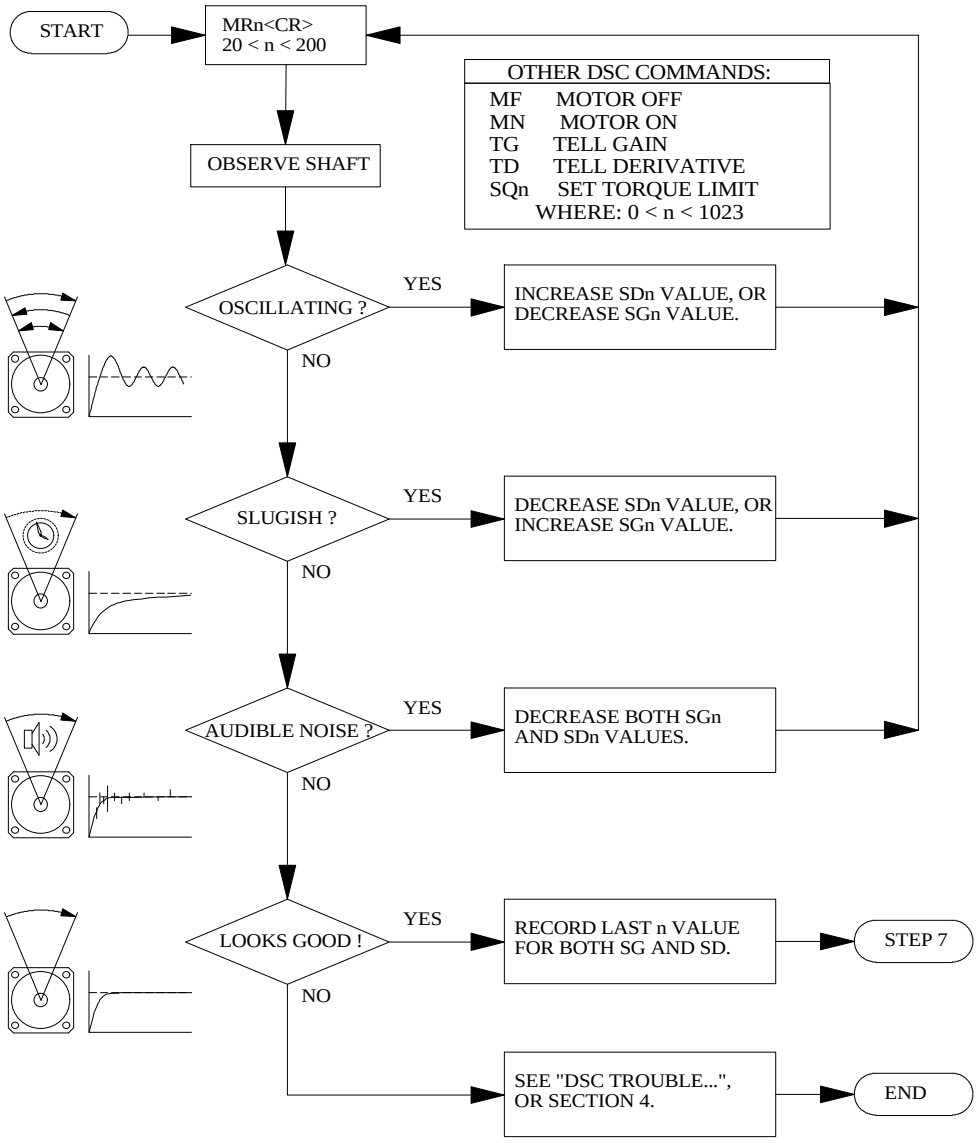
Proportional, Integral, and Derivative parameters are set by entering SG***n***, SI***n***

and SD***n*** commands respectively. In each case, the value ***n*** is a number between 0 and 32000, representing 0% to 100%.

The SG command (set proportional) sets a "spring" like force, which causes the servo to snap to position, but also causes the servo to oscillate. The SD command (set derivative) sets a "shock absorber" like force, which reduces oscillation, but also limits the rate of motion. The objective of the flowchart below is to strike a balance or ratio between SG and SD, while increasing over all control stiffness:

! Important: Initialize the DSC by typing GM<*enter*>, SQ500<*enter*>, and then SI0<*enter*>. SQ500 limits maximum torque to about 50% of rated and SI0 turns the integrator off. When tuning, MR***n*** values should remain small, 20 < ***n*** < 200. The following tuning procedure can result in violent servo motion. Be prepared to use the disconnect or E-Stop device. Make sure that the motor is securely mounted.

SERVO P & D TUNING



Step 7: Storing tuning values in the DSC2000.

The SG and SD values developed in Step 6 can be stored in DSC Flash memory as a "macro" routine. A macro is a string of DSC commands, preceded by a "Macro Define" command, MD***n***, and terminated with carriage return character. Commands within the string are separated by comma characters.

Up to 255 macros can be stored in the DSC. The MD "***n***" value refers to the individual macro number. Macro number 0 is located in Flash memory and is used to embed programs in the DSC. See section 3 for additional information. Do not use macro 0 at this time

Macro numbers 1 through 9 are located in RAM memory. These macros are available for "scratch pad" use when developing a DSC application. RAM based macros can be programmed multiple times without resetting memory. This is accomplished by simply retyping the macro string. However, RAM based macros are not battery backed, and will be lost if logic power is removed from the DSC.

Macro numbers 10 through 255 are located in Flash memory. These macro can only be programmed one time, unless flash memory is reset using the RM command. Before programming a Flash based macro, the motor must be disabled by entering the MF command. In the example below, TM-1 is used to identify any macros that are already programmed:

Example:

```
>TM-1<Enter>
MD255,PM,SG500,SD4000,SI0,...     {macro 255 is used.}
>
                          {-or-}
>                                 {Just ">", no macros used.}
>MF<Enter>                        {Turn motor OFF}
>MD254,SG500,SD2000,SI0<Enter>    {Def. Macro #254 as...}
>TM254<Enter>                     {Check to see if stored.}
MD254,SG500,SD2000,SI0            {TMn returns macro n.}
>MC254<Enter>                     {Call macro number 254.}
>TG,TI,TD<Enter>                  {Tell new PID value.s}
0000000500
0000002000
0000000000
>                                 {MC254 was executed !}
```

The next time +5 VDC logic power is cycled, the DSC will initialize SG***n***, SI***n*** and SD***n*** to default settings. However, in the above example, MC254 can be entered and the "Minimum Setup" tuning values will be restored. See Appendix B for a list off DSC default settings.

The following discussion is provided to familiarize the first time user with basic DSC2000 operation. It describes how to create simple motion using the position and velocity modes of operation. The DSC must be connected and communicating with a PC or terminal device. The servo motor must be connected to the DSC and minimally tuned. If not, see previous sections before proceeding:

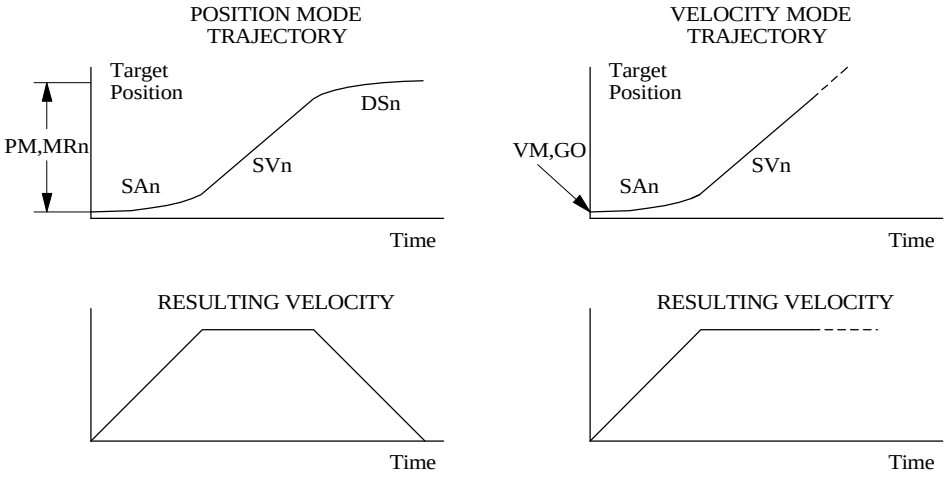
Trajectory Generation.

As mentioned in Theory of Operation, one of the DSC's "background" functions is to generate motion profiles, or trajectories. The trajectory function drives the servo motor from one position to another, by generating desired position 2000 times per second. This desired position is feed into the position PID loop, which in turn, forces the motor to track position.

The DSC2000 provides the user with six different trajectory generation functions to choose from. The user tells the DSC which type to use by entering the appropriate command. Two of these trajectory generation functions are listed below:

Command:	Name:	Function:
PM	Position Mode	Creates "trapezoidal" velocity profile, while positioning the motor from point A to point B.
VM	Velocity Mode	Creates constant velocity by continuously advancing desired position. Creates trapezoidal velocity profile in response to new velocity request.

The word "trapezoidal" refers to the shape of the "velocity vs. time" curve that results from the "position vs. time" trajectory function.



Related DSC commands, SA, SV and DS, provide position "rate of change" information to the trajectory generator. These are "parameter setup" commands. Parameter setup commands are typically entered prior to entering a move or "GO" command. See Command Set Summary for a detailed description of these, and any commands that are used in the following examples.

The set velocity command, SV***n***, limits trajectory "rate of position change" to ***n*** quadrature encoder counts per 0.5 millisecond interval. This 32 bit parameter has a 16 bit integer part, and a16 bit fractional part. SV***n*** is calculated as follows.

Example: Given a 1000 line encoder, determine ***n*** for a limit of 1500 rpm.

1500 rpm / 60 seconds per minute = 25 rps
25 rps * 4000 quadrature counts per revolution = 100000 cps
100000 cps * 0.0005 seconds per interval = 50 counts per interval
(Now shift decimal point 16 bits)
50 * 65536 = 3276800 (i.e. 50.0000 counts per interval)
n = 3,276,800

Set acceleration, SA***n***, and deceleration set, DS***n***, limit the "rate of velocity change". Once SV***n*** is known, acceleration and deceleration values can be calculated based on desired time.

Example: Given SV3276800, determine ***n*** for a 100 millisecond acceleration or deceleration period.

100 milliseconds / 0.5 milliseconds/interval = 200 intervals
3276800 / 200 intervals = 16384
(i.e. 0.2500 encoder counts per interval, per interval, ramp rate)
n = 16,384

All three parameters are entered at the DSC prompt as follows:

```
>SV3276800<Enter>        {velocity}
>SA16384<Enter>          {acceleration}
>DS16384<Enter>          {deceleration}
>
```

Initiating motion in position mode.

In position mode, two different commands can be used to initiate motion. Move relative, MR***n***, tells the DSC to initiate motion to ***n*** encoder counts from present motor position. MA***n*** tells the DSC to position the motor to ***n*** encoder counts relative to position 0. In either case, the value ***n*** is an integer number between -2,147,483,647 and +2,147,483,647.

Example: Move to position 40000.

```
>PM<Enter>               {switch to position mode}
>MN<Enter>               {enable motor, "motor on"}
>DH<Enter>               {define present position as 0, home.}
>MA40000<Enter>          {move to absolute position 40000}
>MR-40000<Enter>         {move to relative -40000 to position 0}
>
```

Initiating motion in velocity mode.

Velocity mode operation, VM, is similar to position mode operation. SA, SV, and DS commands are used to shape the trajectory function. However, the set velocity parameter, SV***n***, sets the "magnitude" of velocity only. In velocity mode, the direction command, DI***n***, is used to set travel direction.

Initiating motion is also different. In velocity mode, the GO command is used to initiate motion. After GO is executed, the trajectory generator will accelerate the motor to the set velocity, and hold this velocity, until commanded to change or stop. Velocity may be changed at any time by simply entering a new SV***n*** command. The stop command, ST, is used to terminate motion.

Example: Go, change velocity, and stop.

```
>VM<Enter>                    {switch to velocity mode}
>DI0<Enter>                   {set direction for velocity}
>GO<Enter>                    {start motion in velocity mode}
        {wait for motor to reach speed....}

>SV100000<Enter>              {change to low speed}
        {wait for motor to reach new speed....}

>ST<Enter>                    {stop motion in velocity mode}
        {wait for motor to stop....}

>PM<Enter>                    {switch back to position mode}
>SV3276800<Enter>             {restore original velocity setting}
>MA0<Enter>                   {move back to position 0}
>
```

The following is a quick reference to DSC Status codes, Command Interpreter error codes, and commonly encountered problems. For a detailed summary of internal DSC fault logic, and recovery sequences, reference Appendix D.

DSC Status Code Summary

Status code	Condition	What to Do?	Additional Reference
b.	+5 VDC logic brown out.	1. Check power supply adjustment +5.0 to + 5.1 VDC. 2. Check power supply regulation under load. 3. Check wire terminal connections.	Section 6, Logic Power Requirements.
P.	Processor - Watch dog	1. Check user program for commands that manipulate DSC physical memory or Servo Table data.	Section 3, Memory Organization.
r.	Reset input active	1. Check wiring at P3-3 and P3-4	Section 6, Isolated I/O Port, Group 1 Inputs.
F.	Excessive Following error. (servo poorly controlled)	1. Check encoder feedback phasing. 2. Check motor commutation phasing. 3. Evaluate servo PID tuning. 4. Increase SE parameter value. 5. Check SA, DS, SV, and SQ command parameter values.	Appendix F. / Section 4, Suggestions for Servo loop tuning. / Section 5.
t.	Motor over temperature	1. Check wiring at P2-14.	Section 6, Encoder Port Specifications.
A.	Servo Amplifier fault.	1. Check AC line input voltage, P5. 2. Check DSC base plate temp., >50 deg.C?, allow time to cool. 3. Power off, check for shorted motor phase, or motor power cable. 4. Check current loop settings, SW1-SW-4 switch settings.	Section 6, DSC installation, electrical
C.	Command interpreter fault.	1. Check syntax of entered command. 2. Check parameter range of entered command. 3. Disable Command Error feature by entering CE0 at DSC prompt.	Section 3, Command Interpreter
L.	Limit detected.	1. Check power supply adjustment +5.0 to + 5.1 VDC. 2. Check power supply regulation when loads are introduced and removed. 3. Check wire terminal connections.	Section 5, Command Summary: LM, LN, and LF.
8	Normal Status after reset.	1. Note, DSC is enabled when decimal point turns off.	Section 3, Command Interpreter

Command Interpreter Error Codes

Code	Condition	What to Do?
?-1	Parameter error.	See section 5, Command Summary, for allowable "***n***" parameter range by individual command.
?-2	Command not valid.	See Sections 3 and 5, check command syntax.
?-3	Negative repeat.	Negative command parameter not allowed.
?-4	Macro definition command not first.	Macros can not define other macros. Chuck for multiple use of MD command within macro command strings
?-5	Macro number out of range.	0 to 255 allowed. Check MD***n*** parameter.
?-6	Macro does not exist.	See MJ and MC commands. Macro number must exist when called.
?-7	Command canceled.	"Esc" character received at serial port. Command execution is halted.
?-8	Axis number out of range.	Check command syntax.
?-9	Motor on not allowed during command.	Motor must be disabled using MF command before programming Flash based macros.

Common problems

Serial port operation

Serial Port not responding, DSC echoes characters only.	Press the "Esc" key to halt command execution. If the DSC is busy executing commands, it will echo any characters that it receives at the serial port, but will otherwise ignore them when but busy.

Command execution

Command was ignored.	Check command usage by trajectory mode, section 5. For example, the MR command is ignored when the DSC is in velocity mode (VM).
Issuing two consecutive moves, only last move executed.	When the DSC executes a move command, it initiates motion, and then continues to interpret any commands that follow. If a new move command is entered before the present move is complete, the DSC will abort the present move and initiate the new move. Try separating move commands with the Wait for stop command, WS, or try polling DSC status register, TS, before issuing a new move.
Using multiple repeat commands	The DSC does not support nested looping within a single command string. Nested loops can be created by nesting macro calls, MC, and programming one loop, (RP, etc.) per macro.
Hex mode/Decimal mode.	The DSC is case sensitive when interpreting HEX numbers. Use upper case letters for the command parameter ***n.*** See HM and DM in Section 5.

Section 3.0

DSC Command Interpreter

The DSC is equipped with 128 Kbytes of Flash, and 32 Kbytes of RAM memory. The command interpreter, or "resident DSC firmware", uses approximately 32 Kbytes of Flash memory. The balance of Flash memory is reserved for permanent storage of user programmed macros and data. RAM memory is reserved for temporary user macros, system variables and application data.

All DSC memory is sectioned by type. Each type supports a unique mode, or feature of DSC operation.

Flash organization:

Type	Function	Read/Write	Related Commands
Command Interpreter	Resident DSC firmware	No	Not user accessible.
Macro Storage; 0, 10-255.	Permanent user program storage	Write once, block erase	Macro commands, reset using RM command.
Contour Point Data	Trajectory profile(s) storage	Write once, block erase	Learn mode commands, reset using RM command.

RAM organization:

Type	Function	Read/Write	Related Commands
Servo Table	Serves as window between DSC foreground and background activity.	Read only	Parameter setup, and Reporting commands.
User Registers	Temporary user data.	Yes	Register commands
Macro Storage; 1-9.	Temporary user program storage	Yes	Macro commands.
Position Record Point Data	Captures motor position data.	Read only	Reporting commands.

Knowledge of DSC physical memory locations is not required for normal programming and operation. Except for the Command Interpreter, all memory types are structured as tables, where individual elements are identified by a number, index, or offset, ***n***.

All DSC commands are structured as a two letter pneumonic followed by a single numeric parameter. The two letter pneumonic represents a function call to the DSC, and the parameter ***n*** is a value passed to this function.

General form: XX***n*** ;where "XX" is command, and "***n***" is parameter.

Example: MR500 ;"MR" for move relative, ***n*** = 500.

The particular function performed by a DSC command may vary, or may be unavailable, depending on the selected trajectory mode. A trajectory mode is selected when it is entered, or executed by the DSC. When using any other DSC command, reference the command summary, "**Usage:**" header, for description of command function by trajectory mode.

Example: SQ500 ;If entered after entering PM, VM, etc., this command sets a "limit" on maximum torque.

SQ500 ;If entered after entering QM (i.e. Usage = Torque mode), then this command directly sets output torque.

Only one DSC trajectory mode is selected at a time. Trajectory mode commands are as follows. See section 4 for a full description of these commands.

Command	**Trajectory mode.**
PM	Position mode
VM	Velocity mode
GM	Gain mode
QM	Torque mode
SM	Slave mode
CM	Contour mode

Parameter range:

The range of the parameter ***n*** is depended on the associated command and command function. In general this parameter is either a code number, "0, 1, 2, 3...", or it is a byte, word, or long word value. When using any DSC command, reference the command summary, "**Range:**" header, for valid usage of the parameter ***n***.

Example: SE***n*** ;Available range is: 0 <= ***n*** <= 32,767

Some DSC commands, like "MF" for motor off, or "TS" for tell status, comply with the general form above, but the associated parameter ***n*** has no meaning. In this case, the parameter ***n*** may be omitted, or set to zero when entering a command. If omitted, the DSC will append a value of zero to the command.

Example: **>TS**<*Enter*> ;Tell status, or

>TS0<*Enter*> ; tell status "0".

The DSC supports two types of command addressing. Command addressing determines the location of the command parameter ***n.***

Immediate addressing:

When a command parameter ***n*** is directly appended to the command pneumonic, the DSC interprets ***n*** as the given parameter. This type of addressing is commonly used with parameter setup commands to initialized RAM based system variables.

Example: **>SA2500**<*Enter*> ;Set acceleration parameter to 2500.

Example: **>AL2500**<*Enter*> ;Load user register 0, accumulator, with 2500.
>AR100<*Enter*> ;Copy accumulator to user register number 100.

Register indirect addressing:

When a command parameter ***n*** is preceded by an "@" character, the DSC interprets ***n*** as the user register number where the given command parameter is held. This type of indirect addressing is commonly used to create application variables within a user program.

Example: **>AL2500**<*Enter*> ;Load user register 0, accumulator, with 2500.
>SA@0<*Enter*> ;Set acceleration to (value in accumulator).

Example: **>GS**<*Enter*> ;Copy Select Switch value to accumulator.
>AA10<*Enter*> ;Add "10" to accumulator value.
>AR50<*Enter*> ;Save accumulator value in user register 50.
>MJ@50<*Enter*> ;Jump to macro number(value in register 50)

The DSC supports 256 user registers, numbered 0 to 255. User register 0 functions as an accumulator for logical and arithmetic operations. User register 1, is an extension to the accumulator for multiply and divide operations. Register numbers 2 through 255 are general purpose locations..

Any DSC command parameter can be stored in a user register, and operated on if copied to the accumulator. Register indirect addressing works with all DSC commands that have an associated parameter ***n.***

Each DSC command performs a unique function. Section 5, Command Summary, groups DSC commands by function type. DSC commands are grouped for user convenience and ease of reference only.:

Command group	**Function type**
Parameter Setup	PID loop tuning, system and trajectory settings.
Motion	Trajectory mode selection, motion initiation and termination.
Reporting	Report DSC system settings, status, user registers, etc.
Macro	Define, report and invoke DSC macros.
Channel	Operate on digital inputs and outputs.
Sequence	Control user program flow.
Learn Mode	Program flash based point memory.
Register	Operate on user registers.
Miscellaneous	Serial communication settings, Break, Reset, etc.

In some cases, complex DSC functions are supported by multiple commands. For example, the DSC can record real time servo position data. This function is supported with two configuring commands, PE, DY, one initiating command, PR, and three uploading commands, DO, DR, and DQ. All of these commands are "related" to each other, and grouped under the DSC Reporting type.

Related commands may be of different type. For example, SG, which is used to set the PID loop proportional gain, is grouped with the parameter setup commands. The related command, TG, reports the proportional gain setting, and is grouped under reporting commands.

When referencing Section 5, see the list of "**Related Commands**" at the end of each command description.

DSC commands may be entered at the prompt individually, or as command string. Command strings are created by separating individual commands with comma characters, and terminated by appending a carriage return character to the last command. When commands are entered individually, they are executed immediately. When commands are entered in string format, they are received, buffered, and executed when the terminating carriage return character is received.

Example:

```
;       Commands entered individually, or ...

>SA2500<Enter>                  ;Set acceleration rate.
>DS2500<Enter>                  ;Set deceleration rate.
>SV80000<Enter>                 ;Set velocity rate.

;       commands entered in string format.

>SA2500,DS2500,SV80000<Enter>   ;Set accel., decel. and vel.
```

When commands are entered in string format, simple program loops can be created using the repeat string command, RP.

Example:

```
;       This program reports servo position, waits 250 milliseconds, then loops
        99 times for a total of 100 readings..

>TP,WA500,RP99<Enter>           ;Tell position, wait , repeat string.
```

Program flow, or "branching", within a command string is also supported. See section 5, Command Summary - Sequence commands.

The DSC supports the use of macro commands. A macro command is a single user defined command which represents an entire string of DSC commands. Macros are created by inserting a macro define command, MD***n***, in front of a command string. When MD***n*** is executed, the associated command string is saved in memory as macro number ***n***.

Example: Command string defined as macro number 5.

>**MD5,SA2500,DS2500,SV80000**<*Enter*> ;Macro 5 is set accel., decel. and vel.

Related commands, TM for tell macro and MC for macro call, are used to verify and execute macros.

Example:

>**TM5**<*Enter*>	;Tell macro 5.
MD5,SA2500,DS2500,SV80000	;Macro number 5 is returned.
>**MC5**<*Enter*>	;Set accel., decel. and vel.

The macro call command is generally used for nesting program loops, or creating subroutine functions. When a macro is "called" the DSC retains a record of the present command string, and position within this string. After executing a called macro, the DSC returns to the command immediately following the initiating macro call. If no command follows, then the DSC transmits a prompt character, and waits at the serial port for new commands.

In the above example, a macro jump command, MJ, can be substituted for the macro call command. Unlike macro call, macro jump retains no record of present string and command position. See Section 5 for additional Macro Commands and information.

Saving Macros

Macros are saved when they are entered using the MD command. Macro numbers 1 through 9 are in RAM memory. These macros are provided for temporary scratch pad use. RAM macros can be modified at any time by simply entering a new MD***n*** command string.

Macro numbers 0, and 10 though 255, are in Flash memory. These locations are reserved for permanent program storage. Before saving a Flash based macro, the servo must be disabled using the MF command. Also, Flash based macros can only be programmed once unless Flash memory is reset using the RM command.

Use the following procedure to modify Flash macros:

1) Enter TM-1 at the DSC prompt. A list of all presently defined macros will be returned.
2) Save this list as a text file. With most terminal emulators, this can be accomplished using a communications log. If using Windows, Copy and Paste the list from the Terminal to Note Pad, and save it as a text file.
3) Enter RM at the DSC Prompt. This will delete all programmed macros and initialize DSC memory.
4) Using a text editor, or Note Pad, modify the macro list as required.
5) Down load the modified list. If using windows, configure text transfers to "one line at a time", and Copy list from Note Pad back to the Terminal window.

Macro 0 is reserved for embedding user programs in the DSC. When logic power is applied to the DSC, the command interpreter initializes system and servo parameters to default settings. After initialization, the command interpreter then attempts to execute macro 0. If no commands are found, the command interpreter transmits a prompt character and waits at the serial port for new commands.

Defining macro 0 as a string of commands that initializes and then calls a continuous program loop, allows the DSC to operate in a user defined, stand alone fashion. The DSC command interpreter will continue to execute user commands until no more are available, or until a break command, BK, is encountered.

When busy executing commands, the DSC will echo characters received at the serial port, but will otherwise ignore most serial communications. User program execution can be terminated by transmitting an ESC character (ASCII code 27), or paused by transmitting a SPACE character (ASCII code 30). See Point to Point and Multidrop Serial Communications for additional information.

Example: The following is provided to demonstrate the typical structure of an embedded user program.

```
>MD0,CE1,MJ50<Enter>                    ;Enable Cmd Errors, jump to main loop.
>MD50,MC60,MC70,MC80,RP<Enter>          ;Main loop, call 60, 70 and 80, repeat.
>MD60,DF5,UD0<Enter>                    ;Do if IN1 off - Status display off.
>MD70,DN5,UD255<Enter>                  ;Do if IN1 on - Status display on.
>MD80,RL0,IS16,TS,BK,NO<Enter>          ;If error bit, report & break, else return
```

Reference Section 5, Command Summary, for a detailed description of individual commands above. After the DSC is reset, macro 0 is the starting point for program execution. The CE1 command enables command interpreter errors. This feature provides protection against typographical errors or corrupted DSC memory. Additional configuration or initialization commands are typically found in macro 0. Macro 0 ends with a jump to macro 50.

Macro 50 represents the main program loop. Macro 50 calls macro 60, 70 and 80, and then repeats. Macros 50 and 60 incorporate "Do if on" and "Do if off" commands. This structure provides an efficient method for directing program execution based on the state of individual DSC inputs. Within macro 60 or 70, the update display command, UD***n***, could be replaced with a string of DSC commands that initialize parameters and, or initiate servo motion.

Macro 80 above loads the accumulator with Servo Status information (RL0), and breaks program execution if the motor error bit is set, (IS16). Incorporating a status check routine into the main user program loop, allows the user program to monitor the state of DSC background functions and servo activity.

The DSC supports RS232 point to point, RS485 point to point, and RS485 multidrop serial communication modes. Communication modes are selected as follows:

Mode	Terminal P1-8	Node Switch 5	Interface cable
RS232 point to point	No Connection	OFF	See Section 7
RS485 point to point	Connect to P1-3	OFF	See Section 7
RS485 multidrop	Connect to P1-3	ON	See Section 7

The default DSC2000 communication settings are follows.:

Baud Rate	Parity	Data Bits	Stop Bits	Flow Control
9600	none	8	1	none

Summary of DSC2000 communication related commands:

Command	Name	Function
BR*n*	Baud Rate	The code number ***n*** determines the baud rate selected.
EF	Echo off	Disables echoing of received characters.
EN	Echo on	Enables echoing of received characters. Default DSC setting is EN.
PC*n*	Prompt character	The value of ***n*** determines the ASCII prompt character. Default is ">", ASCII code 62.
XF*n*	X-Off	Enables software handshaking, sets X-Off ASCII code.
XN*n*	X-On	Enables software handshaking, sets X-On ASCII code.

Point to Point Communications

In point to point mode, the DSC supports a simple "speak when spoken to" type protocol. Individual commands, or command strings, are buffered as they are received, and executed when an carriage return character is received. Unless disabled by the EF command, all received characters are echoed back to the host device for verification.

If a reporting type command is issued to the DSC, the DSC will transmit a value back to the Host, followed by a carriage return, and then a prompt character. If the issued command is of any other type, the DSC will simply transmit a carriage return character, followed by a prompt character when execution is complete.

Since the DSC requires a finite amount of time to execute commands, use of X-On / X-Off software handshaking, or host polling for a prompt character to determine DSC readiness, is recommenced. When busy executing commands, the DSC will acknowledge that it is on-line by echoing received characters, but will otherwise ignore, and retain no record of serial port communications.

Improper use of a DSC command, or parameter, may yield a command interpretation error. The DSC transmits an question mark ("?") character, followed by a dash ("-") character, and then code number to indicate that a command interpretation error has occurred. See Section 5, tell error command, TE, for a list of command interpreter error codes.

Up to 15 DSC controls can be networked from a single RS485 terminal or host device. To configure a DSC control for multidrop operation, wire terminal P1-8 to P1-3, set Node Switch 5 to "ON", and reference section 7 for recommended cabling.

In multidrop mode, Node switches 1 through 4 assign a network address, binary 1 through 15, to each DSC control. Node switch 1 is the least significant bit, and Node switch 4 is the most significant bit. Address 0 (i.e. 1 through 4 set to "off") is reserved as a global address code.

Like point to point mode, DSCs configured for multidrop will respond to a reset event by attempting to execute macro 0. If no commands are present, each DSC will then wait at the serial port for new commands. Unlike point to point mode, however, networked DSCs will listen for transmission of a valid address code before responding to any issued commands.

When properly addressed, an individual DSC control will then communicate with the host device as though it were configured for point to point operation, while all other networked DSC controls continue to listen for a valid address code. If a new address code is transmitted from the host, then the previously addressed DSC will drop off line and listen, while the addressed DSC responds to issued commands.

Address codes are formed by adding 128 to the Node setting of each control. This sets the most significant bit of a byte wide transmission, making each code a non ASCII character. Transmitting a code of 128, address equal to 0, selects all networked controls and enables global communication. When communicating globally, all controls will respond to issued commands, but character echoing and prompt character transmission are disabled.

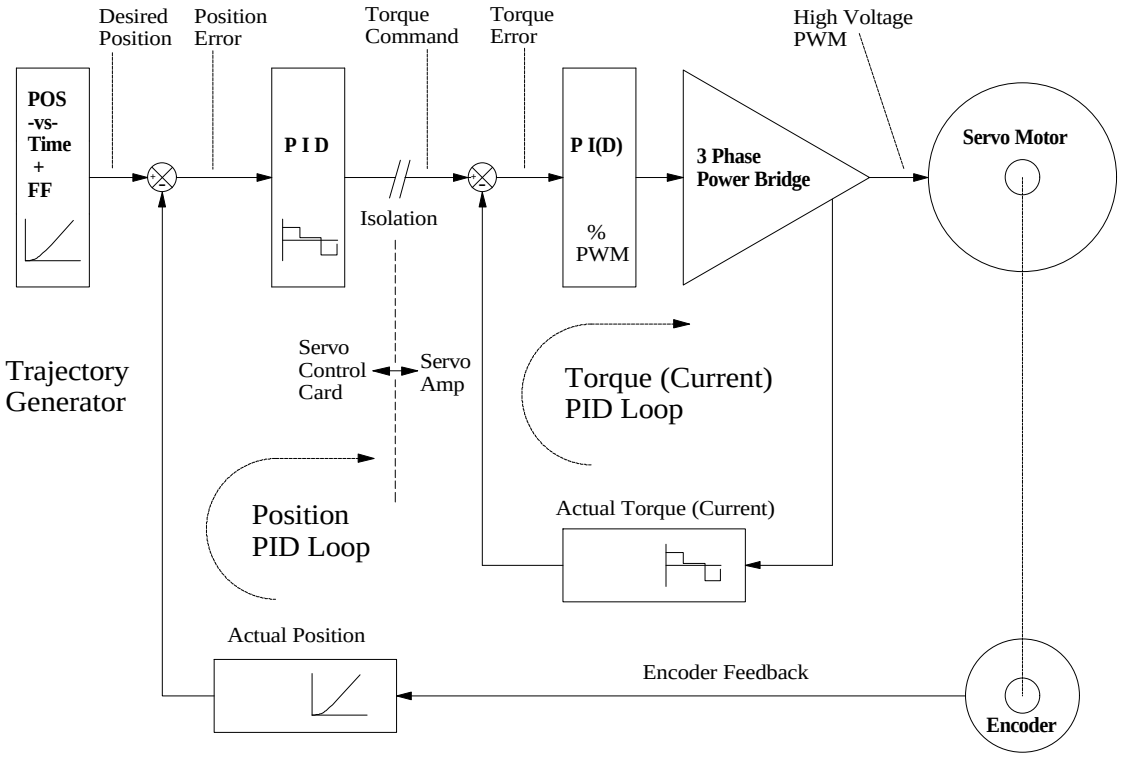
Network Address	**Node switch settings:** **SW1 SW2 SW3 SW4 SW5**	**Transmission code**
Global	XX XX XX XX XX	128
Axis 1	ON OFF OFF OFF ON	129
Axis 2	OFF ON OFF OFF ON	130
Axis 3	ON ON OFF OFF ON	131
Axis 4	OFF OFF ON OFF ON	132
Axis 5	ON OFF ON OFF ON	133
Axis 6	OFF ON ON OFF ON	134
Axis 7	ON ON ON OFF ON	135
Axis 8	OFF OFF OFF ON ON	136
Axis 9	ON OFF OFF ON ON	137
Axis 10	OFF ON OFF ON ON	138
Axis 11	ON ON OFF ON ON	139
Axis 12	OFF OFF ON ON ON	140
Axis 13	ON OFF ON ON ON	141
Axis 14	OFF ON ON ON ON	142
Axis 15	ON ON ON ON ON	143

Section 4.0

DSC Servo Control

The DSC divides processing time into foreground and background activity. In the foreground the DSC interprets user commands. In the background, the DSC controls the servo motor.

DSC Control Diagram



Background processing is transparent to the user. This function is interrupt driven and occurs 2000 times per second. The DSC requires about 0.25 milliseconds to service each background interrupt, which leaves about 50% of the available processing time for foreground operation.

In the background, processing is divided into three fundamental steps. First, "desired position" is calculated by executing the selected trajectory generator function. Second, "position error" is calculated by subtracting "actual position" from desired position. Finally, a "torque command" is generated by executing a PID algorithm that operates on position error.

The servo amplifier completes the motor control function by closing a second control loop on motor torque. This inner loop operates on the torque command generated by the position loop. The servo amplifier controls motor torque (i.e. current) independent of motor speed, or direction. This allows the outer loop to operate on motor load disturbances directly.

The amplifier calculates "torque error" by subtracting actual torque from commanded torque. Using a PID algorithm, the amplifier then calculates the level, or % on time, of Pulse Width Modulation required for elimination of torque error.

Permanent magnet brushless motors offer a linear relationship between motor current and motor torque. The amplifier based current loop controls the level of Pulse Width Modulation applied to the motor. In doing so, the current loop forces motor current to track the position loop torque command. Therefore, the current loop can be thought of as a controlled torque generator. The relationship between the foreground torque setting, or torque limit setting, SQ***n***, and actual motor torque can be quantified as follows:

Example:	SP80A-375AE	DSC20D0-000-0000
	Kt = 5.4 in*lb/A	Ipk = 20 A

Maximum package torque: 5.4 in*lb/A ***** 20 A **=** 108 in*lb.
Package torque sensitivity: 108 in*lb / 1023 ***n*** max = 0.10 in*lb/***n*** cnt.

As with all closed loop systems, the DSC current loop requires tuning to function optimally. This current loop incorporates a Proportional, Integral, Derivative algorithm, and is tuned by setting the gain level of these parameters. Specifically, the D gain is internally fixed at a level that is appropriate for SP and most compatible servo motors. The Proportional and Integral gains can be adjusted by setting the respective SW-1 to SW-4 switch banks located adjacent to the DSC name plate label.

Optimal current loop tuning is most affected by line voltage source, 115 or 230 VAC, motor inductance, Lm, and the motor resistance, Rm. Since these parameters are independent of any mechanism connected to the motor shaft, tuning the current loop effectively "mates" the DSC to a given motor. Once current loop tuning is established, the motor can generally be connected to any load without having to change these settings.

Proper current loop tuning must be established before attempting to tune the outer position PID loop.

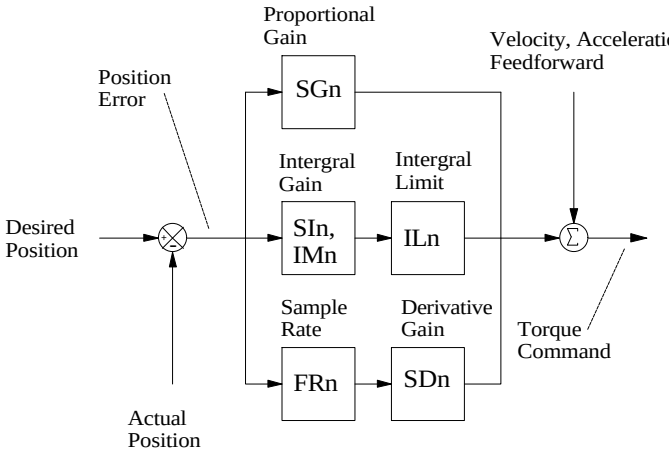
If using an Automotion SP servo motor:

Reference Appendix E for a predetermined list of current loop switch settings. If audible noise (6 to 9 KHz chopping sound) occurs, or the load connected to the motor shaft is both low inertia, and low friction, then current loop tuning may need to be further optimized. Consult Automotion for assistance, and, or reference Appendix G for further information.

If using a compatible servo motor:

Compare the motor's phase to phase inductance and resistance specifications to SP motor specifications. Find the SP motor that most closely matches and use the recommended switch settings in Appendix E. Also, consult Automotion for assistance, and, or reference Appendix G for further information.

When the servo motor is enabled, the position PID loop is executed by the DSC. This loop is structured as follows:



$$Q(t) = SG*E(t)+[SI*\sum E(t) \text{ or } IL]+SD*[E(t)-E(t-FR)]+\{feedforward...\}$$

Where: Q(t) is the Torque Command at time t,
E(t) is the Position Error at time t and
SG, SI, IL, FR, SD are DSC parameter setup commands.

Position error is measured in units of quadrature encoder counts. Position error is sampled and processed through the PID loop 2000 times per second. The output of P, I, and D operators are summed with feed forward terms to produce a torque command. When optimally tuned, this torque command forces the motor to track desired position.

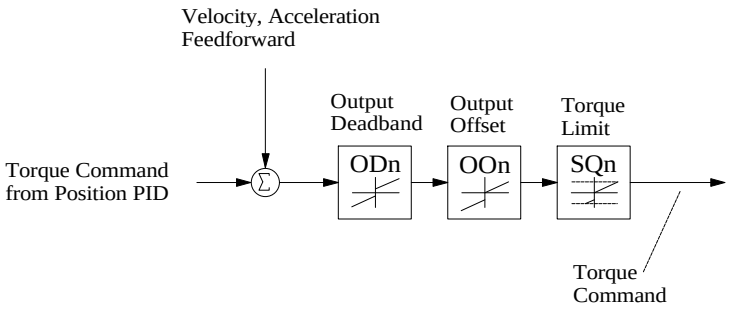
SG sets the proportional gain in this loop. Higher values of SG will increase response stiffness, but may cause the servo to oscillate. SI sets the integration gain in this loop. Higher values of SI will increase response accuracy, but may also cause the servo to oscillate. IM is used to disable the integrator during motion (i.e. when desired position is changing). IL is used to limit the maximum torque that will develop due to integration.

Derivative gain is used to dampen oscillation. Both SD and FR set derivative gain in this loop. FR sets derivative gain by lengthening the time period over which the derivative term is sampled. SD sets derivative gain directly. Higher values of SD and FR will increase stability, but may cause audible noise, or limit the rate of motion during a fast move.

Feedforward terms (velocity, acceleration, and deceleration) are offset values that are added to the PID loop torque command. Feedforward terms are used to minimize position error during a move. These adjustments are "outside" the position loop and do not affect loop stability. Typically, PID parameters are tuned first without the use of feedforward terms. Once PID parameters are tuned, feedforward terms are added to further improve tracking of desired position.

All DSC position loop parameters are programmable This allows the user to optimize servo performance with a specific mechanical load. Unlike feedforward terms, mechanical characteristics of the load are "inside" the loop. Load friction, inertia, and viscous damping, for example, greatly effect the optimal settings for P, I, and D operators. See "Suggestions for Servo Loop Tuning" at the end of this section for additional information.

Following execution of the position PID algorithm, the DSC performs a series of operations which shape the torque command signal before passing it to the servo amplifier.



Output Deadband

Friction in a mechanical system creates torque "deadband". Deadband is the range (typically centered around zero) in which a given level of torque is insufficient to move the load.

The OD command, value ***n,*** is added to the torque output, if the torque output is greater than zero, or subtracted from the torque output, if the torque output is less than zero. This command creates a "jump around zero torque" function which can be used to eliminate the effects of frictional deadband within a given system.

Output Offset

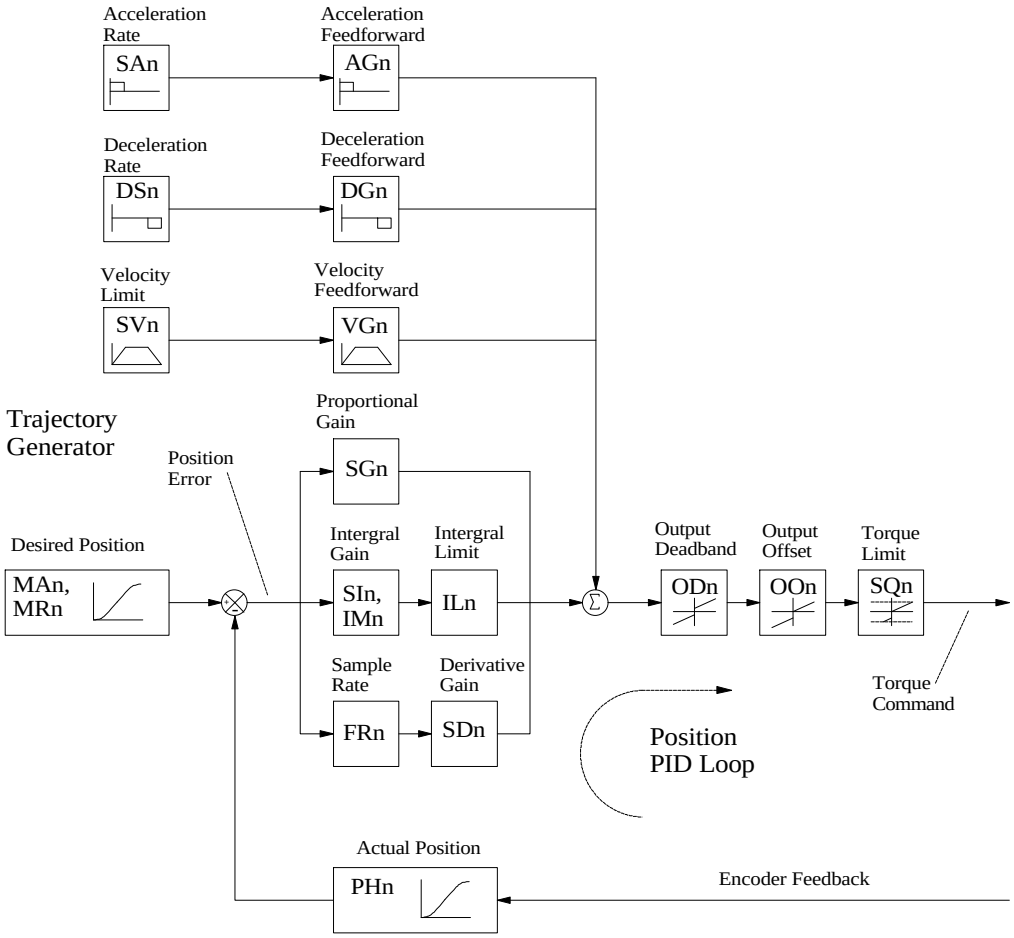
Gravity in a mechanical system creates a the need for a constant level of offsetting torque. This offset is typically maintained by the position loop integrator.

The OO command, value ***n,*** is added to the torque output value after operated on by the OD function. This feature can be used to compensate for a gravitational load by setting a value ***n*** equal to the amount of torque required to hold position against gravity. Use of the OO command can eliminate dependence on the position loop integrator, providing greater accuracy with less potential for instability.

Torque Limit

This command sets an absolute limit on the torque level that is produced by the position PID loop. The value *n* / 10 equals the percentage of maximum torque output for a given DSC package. To find the maximum package torque, multiply the DSC's peak current rating by the motor's torque constant, Kt. This command can be used to protect mechanical systems from physical damage, especially when developing application programs, or tuning the position PID loop.

In position mode, the DSC generates trapezoidal trajectory profiles for point to point positioning of the servo. Position mode is selected by entering "PM" at the DSC prompt. After PM is executed, the DSC background servo control function is configured as follows:



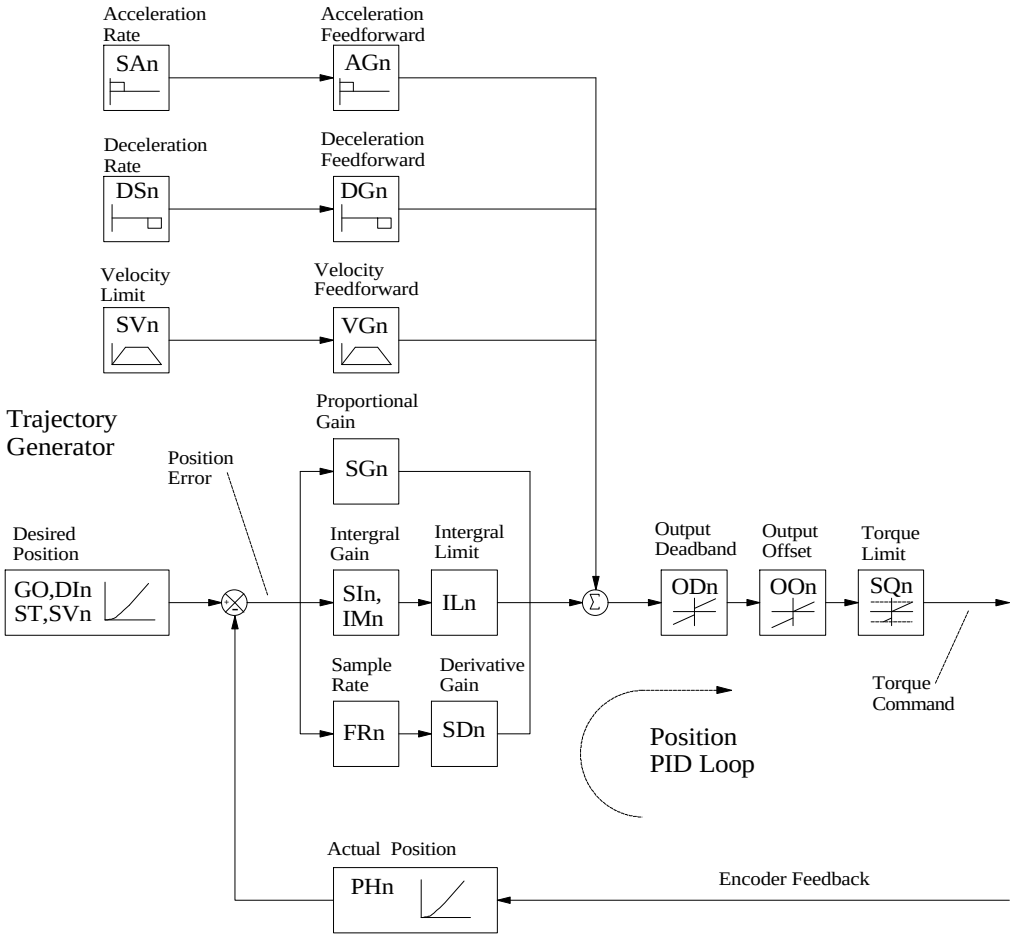
Trajectory Generation

Motion is initiated using the move absolute command, MAn, or move relative command, MRn. When initiated, the trajectory generator advances "desired position" over time based on SA, DS, and SV parameters. The resulting velocity profile is trapezoidal in shape. If insufficient move distance exists, the SV limit will not be reached, and the velocity profile will be triangular in shape, consisting of acceleration and deceleration only.

Loop Structure

All PID, and post PID operators are present. Acceleration, deceleration, and velocity feedforward are also available.

In velocity mode, the DSC generates a trapezoidal trajectory profile for changing servo velocity, and continuously advances "desired position" for maintaining constant velocity. Velocity mode is selected by entering "VM" at the DSC prompt. After VM is executed, the DSC background servo control function is configured as follows:



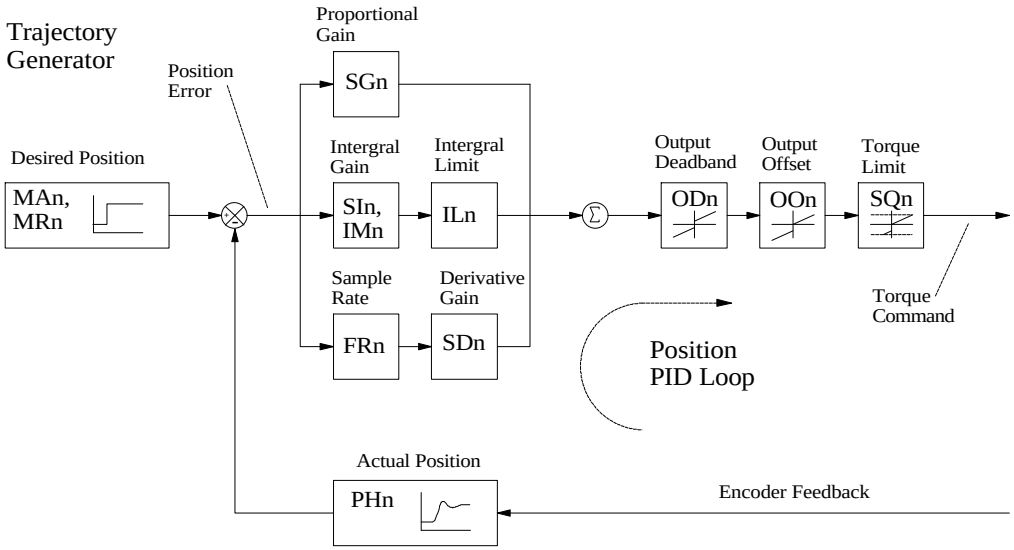
Trajectory Generation

Motion is initiated with the GO command, and terminated with the stop command, ST. In velocity mode, SV sets the magnitude of velocity only, and DI is used to set the direction of velocity. When motion is initiated, the trajectory generator advances "desired position" over time based on SA and SV parameters. Velocity may be changed at any time by issuing a new SV command. If a new velocity setting is less than the present setting, the trajectory generator will advance desired position based on the DS parameter until the new velocity is reached.

Loop Structure

All PID, and post PID operators are present. Acceleration, deceleration, and velocity feedforward are also available.

In gain mode, the DSC generates a non profiled trajectory for point to point positioning of the servo. Gain mode is selected by entering "GM" at the DSC prompt. After GM is executed, the DSC background servo control function is configured as follows:



Trajectory Generation

Motion is initiated using the move absolute command, MAn, or move relative command, MRn. When initiated, the trajectory generator immediately advances "desired position" to the finial end target position, ***n***. The resulting velocity profile and motion is controlled by the PID algorithm alone. This mode of operation provides a position "step" function, and is commonly used to tune PID parameter settings.

Loop Structure

All PID, and post PID operators are present. Feedforward operators are disabled.

In torque mode, the trajectory generator and position PID loop are disabled. Torque mode is selected by entering "QM" at the DSC prompt. After QM is executed, the DSC background function is configured as follows:



Trajectory Generation

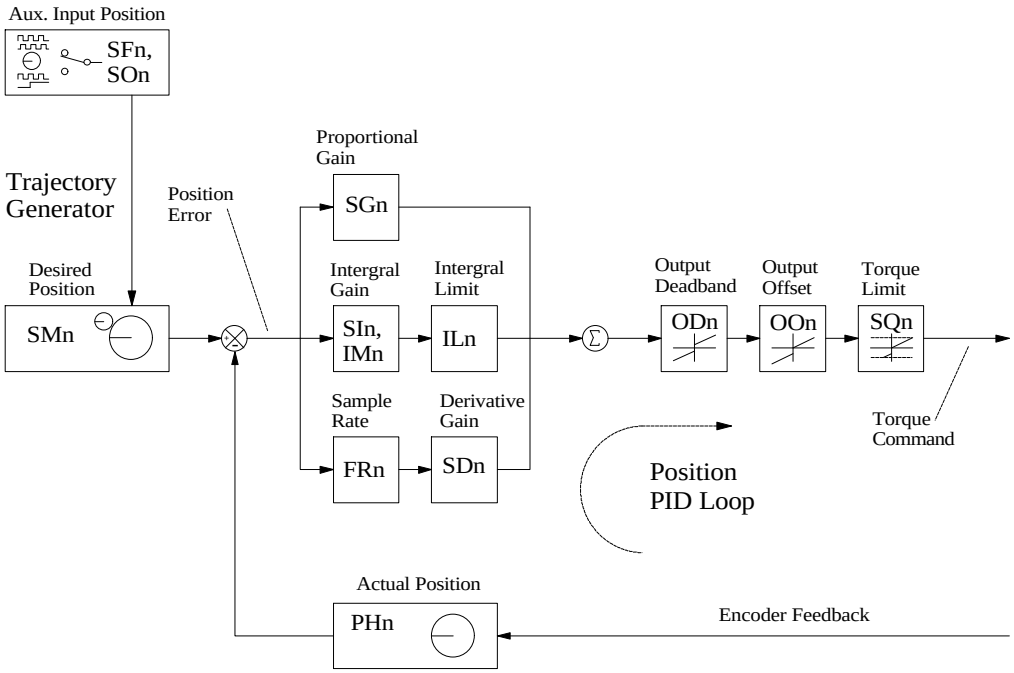
Trajectory generation is disabled. Position may be observed by using the TP command. In torque mode, torque is commanded directly using the SQn command. Given sufficient torque, the servo will accelerate in an uncontrolled fashion to the no load package RPM rating.

Torque mode is commonly used to tune "torque PID loop" parameters. Also, user created control algorithms may use this mode for tensioning, tightening or force related applications.

Loop Structure

All PID, post PID, and feedforward operators are disabled.

In slave mode, the DSC generates motion by tracking the position of an auxiliary encoder, or step and direction input. SM mode is selected by entering "SM***n***" at the DSC prompt. After SM***n*** is executed, the DSC background servo control function is configured as follows:



Trajectory Generation

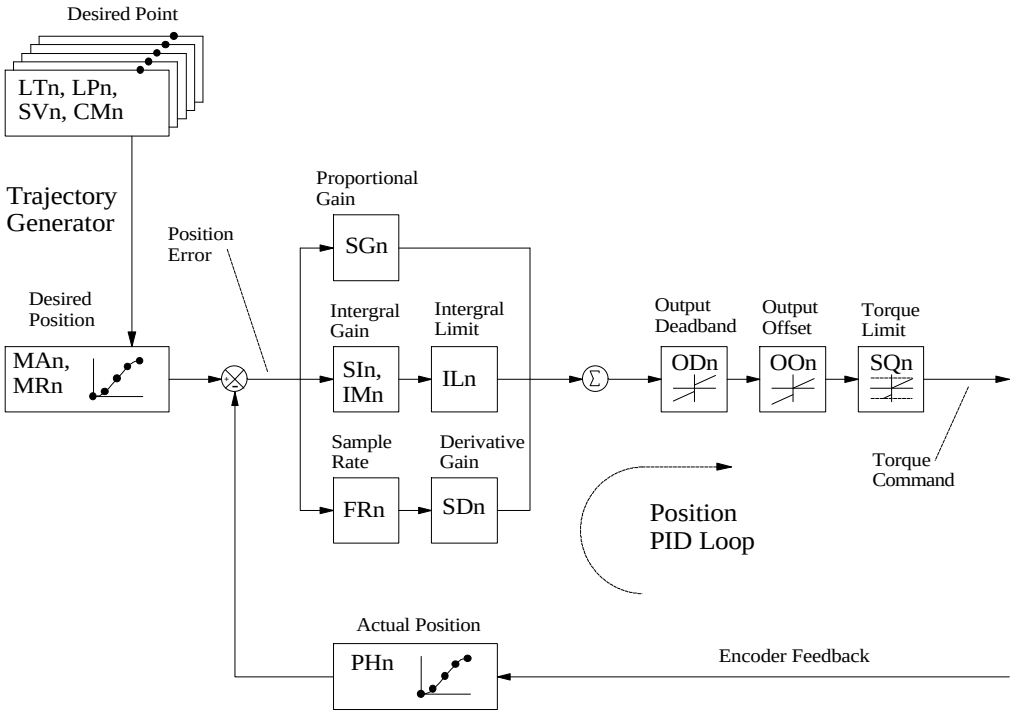
Slave motion is initiated using the SM***n*** command. When initiated, the trajectory generator advances "desired position" over time based on the SM ratio, ***n***, between auxiliary input position and desired position.

The slave format command, SF***n***, determines the relative direction between input position and desired position. SF also selects the auxiliary input format, quadrature encoder or step and direction. The slave offset command, SO***n***, sets a fixed offset value between input position and desired position.

Loop Structure

All PID, and post PID operators are present. Feedforward operators are disabled.

In contour mode, the DSC generates trajectory profiles based on desired contour points stored in Flash memory. Contour mode is selected by entering "CM***n***" at the DSC prompt. After CM***n*** is executed, the DSC background servo control function is configured as follows:



Trajectory Generation

the CM, value ***n***, initializes the contour point index. Motion in contour mode is initiated using the move absolute command, MA***n***, or move relative command, MR***n***, where ***n*** is target point index. When motion is initiated, the trajectory generator advances "desired position" over time by linearly interpolating between contour points. The SV command determines the number of interpolations between points.

Loop Structure

All PID, and post PID operators are present. Feedforward operators are disabled.

DSC Recording Functions

A powerful tool supplied with the DSC is the ability to record position and torque data while the servo is in motion. Once recorded, this data can be uploaded from DSC memory, and imported to a user supplied spreadsheet program for analysis. Recorded data can be used to quantify servo performance as PID loop parameters are adjusted.

Summary of DSC recording commands:

Command	Function	*n* Range	Page
DY***n***	Sets an enable delay period to ***n****0.5 mS.	0 to 32767	5-28
PE***n***	Sets a sample period of ***n****0.5 mS.	0 to 32767	5-29
PR***n***	Enables recording function for ***n*** samples.	0 to 1200	5-29
DO***n***	Displays "optimal" position for sample ***n***.	0 to 1200	5-26
DQ***n***	Displays "torque command" for sample ***n***.	0 to 1200	5-27
DR***n***	Displays "recorded" position for sample ***n***.	0 to 1200	5-27

See "Example" routines given in section 5 which demonstrate the use of these commands. The following 7 graphs can be produced by uploading and manipulating recorded data.

1) "Desired Position vs. Time", graph DO***n*** samples from 0 to ***n***.
2) "Desired Velocity vs. Time", graph [DO***n***-DO(***n***-1)] from 1 to ***n***.
3) "Actual Position vs. Time", graph DR***n*** samples from 0 to ***n***.
4) "Actual Velocity vs. Time", graph [DR***n***-DR(***n***-1)] from 1 to ***n***.
5) "Position Error vs. Time", graph [DO***n***-DR***n***] from 0 to ***n***.
6) "Velocity Error vs. Time", graph [[DO***n***-DO(***n***-1)]-[DR***n***-DR(***n***-1)]].
7) "Torque vs. Time", graph DQ***n*** samples from 0 to ***n***.

If using a PC or compatible, Automotion offers a Windows based tuning utility that provides menu driven test and graph capabilities. Contact Automotion for additional information.

Tuning to Load Characteristics

Page 2-12 of this manual, First Time Operation - Step 6, describes a minimal method for tuning PID parameters. If this testing was performed on a bench top with no load connected to the motor shaft (as recommended), then the PID loop parameters will need to be tuned again, when the motor connected to the intended application load.

Load characteristics such as friction, inertia, viscous damping, can greatly affect optimal setting of PID parameters. Loop parameters appropriate for control of an unloaded motor, may produce unstable or oscillating motion when connected to the load.

! Important: Tuning PID loop parameters with the motor connected to the load can cause unexpected and violent motion. Automotion recommends enabling and testing Limit switch inputs before enabling and tuning the servo. Use of limit switch inputs can protect the load mechanism from damage. See LM, LN, and MN commands in section 5 of this manual.

Also see the "set following error" command, SE, in section 5. Use this command to automatically disable the servo due to uncontrolled motion.

Is the load controllable?

The DSC can control a wide range of load mechanisms. However, some load mechanisms are easier to control than others. Specifically, "ridged body" type loads are preferred.

A ridged body load is any mass or inertia that is tightly coupled to the motor shaft, and not separated from the motor shaft by mechanical backlash, or a compressible member. Mechanical backlash, or the springy behavior of a shaft coupling can greatly complicate PID tuning. If severe enough, these load characteristics may prevent the DSC from controlling position in a stable manner. A load is considered to be "ridged" if backlash and compression is held to within a least few encoder counts.

Many types of shaft couplings, gearboxes, and belt drives are specifically designed for servo driven applications. These specialty components are optimized to eliminate backlash and compression. If required, contact Automotion for assistance and, or component recommendations.

Finding Output Deadband

Friction in a mechanical system creates torque "deadband". Deadband is the range (typically centered around zero) over which a given level of torque is insufficient to move the load.

The OD command, value ***n,*** is added to the torque output, if the torque output is greater than zero, or subtracted from the torque output, if the torque output is less than zero. This command creates a "jump around zero torque" function which can be used to eliminate the effects of frictional deadband within a given system.

To find the proper setting of OD***n***, configure the DSC for position mode by entering PM, and turn the PID integrator off by entering SI0. Enable the servo using the MN command and verify that it holds position in a stable manner. Incrementally increase OD***n*** values from 0, 1, 2 ...., to a level that produces constant shaft dithering or oscillation. Reduce the last OD***n*** value by about 10%, or at least 10 counts, and use this value as a final setting.

Finding Output Offset

Gravity in a mechanical system creates a the need for a constant level of offsetting torque. This offset is typically maintained by the position loop integrator. The OO command, value ***n,*** is added to the torque output value after being operated on by the OD function. Use of the OO command can eliminate dependence on the position loop integrator, providing greater accuracy with less potential for instability.

To find the proper setting of OO***n***, configure the DSC for position mode by entering PM, and turn the PID integrator off by entering SI0. Enable the servo using the MN command and decrease the proportional gain setting, SG, to a level that maintains stability, but otherwise results in loose control of position. Once SG is lowered, incrementally increase or decrease OO***n*** values until following error, TF, is eliminated. When OO***n*** is determined, SG and SI values can be restored.

IF the D term is adjusted too High

When tuning the DSC to a load, the general goal is increase loop gains for tighter control. Typically, the proportional gain, SG, is increased until the servo begins to overshoot during a move, or oscillate when holding position.

To compensate for this behavior, the derivative gain, SD, is then typically increased.

This approach to tuning, constantly increasing SD and SG, will eventually result in excessive loop gain. The maximum setting of proportional gain, SG, is limited by the amount of dampening that exist naturally in the load, or that can be created artificially by increasing SD. In turn, the maximum setting of SD is limited by "quantized" position error.

Within the servo loop, position error is calculated by subtracting desired position from actual position. Since position error is measured in units of encoder counts, it is quantized to an integer level; 0, 1, 2, 3,... etc., each time the loop is processed.

If the servo motor is moving at low velocity, or the servo loop is updating at a high rate, then the resolution of error that SD operates on will be small. Under such conditions, increasing SD will eventually cause the servo loop to over compensate by dramatically amplifying the difference between individual encoder counts.

If SD is increased to the point where significant audible noise is present, and yet the load is still poorly controlled, then a higher resolution feedback encoder may be required. Contact Automotion for recommendations.

Adjusting D for a High Inertial load.

As a general rule in control theory, the rate at of response from a control loop should be higher than the rate at which a load can change. This is a simple way of saying that a control loop's bandwidth should be kept as high as possible.

The DSC updates the position PID loop 2000 times per second, giving it an effective bandwidth of about 400 Hz. Most mechanical loads are incapable of moving or oscillating at frequencies above 200 Hz. The high bandwidth of the DSC position loop, allows the DSC to sense and correct for oscillation as it occurs.

High inertial loads, like flywheels, are incapable of oscillating at high frequency. This means that they are naturally stable when running at constant speed, and require very little loop gain to control. However, because they oscillate so slowly, the derivative gain, which operates on the rate of change of error, may be incapable of sensing oscillation. Hence, the derivative gain may be unable to stabilize the load.

To correct for this condition, the DSC PID algorithm provides a separate time base, or sample period, for adjusting derivative gain. This period is set by using the FR***n*** command. If the ratio of load inertia to motor inertia exceeds 20:1, try increasing the derivative sample period when tuning to the load. This method of tuning can accommodate inertia ratios beyond 2000:1.

IF the I term is adjusted too High

The PID integral gain setting, SI, provides position control accuracy. This term operates on the persistence of position error by increasing the torque command over time until error is eliminated. However, integral gain is also a source of loop instability. In general, to avoid servo oscillation, the ratio of derivative gain to integral gain should be kept high; 100:1 or more.

To minimize the need for integral gain, adjust output deadband, OD, and output offset, OO, before adjusting SI. Use the integration limit command, IL, to prevent servo excessive torque levels due to integration. Also, use the IM command to disable the integrator during motion if not required. See Section 5 in this manual for further information these commands.

Updating P and D on the fly.

There is no one correct set of PID parameters that is better than all others. Tuning a PID loop is like trying to solve 3 mathematical equations containing 4 unknown variables. The answer is not a single set of values, but rather a family of values. Optimal tuning can only be achieved when some additional equation, or application requirement is taken into account.

For example, if holding position tightly when stationary is more important that tracking position during a move, then the PID parameters can be optimized for this requirement. If getting from point A to point B quickly is more important than holding position accurately, then the PID parameters can be optimized for this requirement.

DSC loop parameters can be updated on the fly. This means that optimal PID values can be determined for each unique application requirement. Once determined, these values can be updated before, during or after a move.

Tuning with trajectories

Page 2-12 of this manual, First Time Operation - Step 6, describes a minimal method for tuning PID parameters. The DSC was configured for gain mode operation and a position step function was generated to excite the load. Once excited, the PID loop response was evaluated by observing the motor shaft.

When fine tuning PID loop parameters to the application load, try exciting the load with the actual trajectory modes, and motion profiles that will be used in the intended application. Step response testing is good starting point, but will not necessarily yield the optimal PID settings for all situations.

Position and Velocity modes offer feedforward terms. When properly adjusted, feedforward terms provide open loop accuracy by commanding torque directly, rather than depending on the PID loop to produce it. If using feedforward terms, PID parameters can be optimized for alternate application requirements.

Check for unconditional stability.

Once application specific PID parameters are determined, Automotion recommends testing the servo system to verify that it is unconditionally stable. Unconditionally stable means that the selected PID parameters prevent the servo from bursting into uncontrolled oscillation when the load, or control loop, is excited at some resonant frequency.

Although analytical methods exist for verifying unconditional stability, these methods are seldom practical. An imperial method that can be used is to create a large scale step function in gain mode. A step function is a square wave input that carries with it a range of higher order odd harmonic frequencies. Large scale means that the current command is allowed to saturate at maximum during the test.

Note that contour mode operation, CM, can be used to generate alternate driving functions. For example, a triangle function which carries with it higher order even harmonic frequencies can be used. Contact Automotion for assistance or further explanation if required.

Section 5.0

DSC Command Summary

Parameter Setup Commands

AGn	Acceleration gain	5-5
DBn	Target deadband	5-5
DDn	Direction Deadband	5-6
DGn	Deceleration Gain	5-6
DSn	Deceleration Set	5-7
FRn	Set derivative sample period	5-7
ILn	Set integration limit	5-8
IMn	Integration mode	5-8
LFn	Limit switch off	5-9
LMn	Limit switch mode	5-9
LNn	Limit switch on	5-10
ODn	Output deadband	5-10
OOn	Output offset	5-11
PHn	Set servo output phase	5-11
SAn	Set acceleration	5-12
SDn	Set derivative gain	5-12
SEn	Set following error	5-13
SFn	Slave input format	5-13
SGn	Set proportional gain	5-14
SIn	Set integral gain	5-14
SOn	Slave offset	5-15
SQn	Set torque	5-15
SVn	Set velocity	5-16
VGn	Velocity gain	5-16

Motion Commands

AB	Abort motion	5-17
CMn	Contour mode	5-17
DHn	Define home	5-18
DIn	Velocity direction	5-18
FIn	Find index	5-19
GH	Go home	5-19
GM	Gain mode	5-20
GO	Start velocity	5-20
MAn	Move absolute	5-21
MF	Motor off	5-21
MN	Motor on	5-22
MRn	Move relative	5-22
PM	Position mode	5-23
QM	Torque mode	5-23
SMn	Slave mode	5-24
ST	Stop velocity	5-24
VM	Velocity mode	5-25

Reporting Commands

AT	Tell auxiliary position	5-26
DOn	Display optimal position	5-26
DQn	Display recorded torque	5-27
DRn	Display recorded position	5-27
DYn	Set record delay	5-28

HE	List available commands	5-28
PEn	Set record period	5-29
PRn	Position record enable	5-29
TAn	Tell analog	5-30
TB	Tell breakpoint	5-30
TCn	Tell channel	5-31
TD	Tell derivative gain	5-31
TE	Tell command error	5-32
TF	Tell following error	5-32
TG	Tell proportional gain	5-33
TI	Tell integral gain	5-33
TL	Tell integration limit	5-34
TO	Tell optimal	5-34
TP	Tell position	5-35
TQ	Tell torque	5-35
TS	Tell servo status	5-36
TT	Tell target position	5-36
TV	Tell command velocity	5-37
TX	Tell contour index	5-37
TZ	Tell servo index	5-38
VE	Tell firmware version	5-38

Macro Commands

JPn	Jump to command absolute	5-39
JRn	Jump to command relative	5-39
MCn	Macro call	5-40
MDn	Macro define	5-40
MJn	Macro jump	5-41
RMn	Reset memory	5-41
TMn	Tell macro	5-42

Channel Commands

BCn	Breakpoint channel	5-43
CFn	Channel off	5-43
CHn	Channel high true logic	5-44
CLn	Channel low true logic	5-44
CNn	Channel on	5-45

Sequence Commands

DFn	Do if channel off	5-46
DNn	Do if channel on	5-46
DTn	Delay at target	5-47
IBn	If accumulator is below	5-47
ICn	If accumulator bit is clear	5-48
IEn	If accumulator is equal	5-48
IFn	If channel is off	5-49
IGn	If accumulator is greater	5-49
INn	If channel is on	5-50
IPn	Interrupt on absolute position	5-50
IRn	Interrupt on relative position	5-51

ISn	If accumulator bit is set	5-51
IUn	If accumulator is unequal	5-52
RPn	Repeat command string	5-52
WAn	Wait for time	5-53
WFn	Wait for channel off	5-53
WNn	Wait for channel on	5-54
WPn	Wait for absolute position	5-54
WRn	Wait for relative position	5-55
WSn	Wait for stop	5-55
WTn	Wait for target	5-56

Learn Mode Commands

LPn	Learn point	5-57
LTn	learn target	5-57

Register Commands

AAn	Accumulator add	5-58
AC	Accumulator compliment	5-58
ADn	Accumulator divide	5-59
AEn	Accumulator exclusive OR	5-59
ALn	Accumulator load	5-60
AMn	Accumulator multiply	5-60
ANn	Accumulator AND	5-61
AOn	Accumulator OR	5-61
ARn	Copy accumulator to register	5-62
ASn	Accumulator subtract	5-62
GAn	Copy analog input to accumulator	5-63
GBn	Copy byte to accumulator	5-63
GLn	Copy long word to accumulator	5-64
GN	Copy node switch to accumulator	5-64
GS	Copy select switch to accumulator	5-65
GWn	Copy word to accumulator	5-65
PBn	Copy accumulator to byte	5-66
PLn	Copy accumulator to long word	5-66
PWn	Copy accumulator to word	5-67
RAn	Copy register to accumulator	5-67
RBn	Servo table byte to accumulator	5-68
RLn	Servo table long word to accumulator	5-68
RWn	Servo table word to accumulator	5-69
SLn	Accumulator shift left	5-69
SRn	Accumulator shift right	5-70
TRn	Tell register	5-70
UDn	Update display	5-71
WBn	Accumulator to servo table byte	5-71
WLn	Accumulator to servo table long word	5-72
WWn	Accumulator to servo table word	5-72

Miscellaneous Commands

BK	Break	5-73
BRn	Baud rate	5-73

CAn	Convert register to ASCII	5-74
CEn	Command error enable	5-74
DM	Decimal mode	5-75
EF	Echo off	5-75
EN	Echo on	5-76
HM	Hex mode	5-76
NO	No operation	5-77
PCn	Prompt character	5-77
RT	Reset MPU	5-78
XFn	X-off code	5-78
XNn	X-on code	5-79

AGn

Acceleration gain

This command sets the acceleration feed forward gain for the servo. This feed forward produces a torque value proportional to the set acceleration rate, when the servo is accelerating. This torque value is numerically summed with the output of the PID algorithm.

Once this value is set, acceleration rates may be changed, and the corresponding feed forward value will be determined automatically. See "Suggestions for Servo Loop Tuning" in this manual for additional information.

Range: -32,767 <= ***n*** <= 32,767

Registers: Servo Table, offset 78

When the AG is executed, the value ***n*** is stored in the RAM based Servo Table at offset 78, decimal. The value ***n*** can be accessed by executing RW78,TR0<*CR*>. The default value is 0.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - No
Contour Mode - No

Example:

```
;     This routine causes the acceleration rate to
;     change while the servo is in motion.

DH,+          ;Define as home.
AG250,+       ;Set accel gain to 250
SA500,+       ;Set accel to low value
MR40000,+     ;Start move
WP4000,+      ;Wait for position 4000.
SA25000+      ;Now set accel to high.
```

Related Commands: **SA, DS, DG**

DB*n*

Target deadband

This command sets bit 2 in the servo status register when the servo's actual position is within +/- *n* encoder counts of the final target position. Bit 2 of the servo status register can be evaluated as required within a user's program. In this way, the user can determine that a move has been completed to practical level of precision.

See Register and Sequence commands for further information.

Range: 0 <= ***n*** < = 32,767

Registers: Servo Table, offset 102
Servo status, bit 2

When the DB command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 102, decimal. The value ***n*** can be accessed by executing RW102,TR0<*CR*>. The default value is zero. Bit 2 of the Servo status register can be viewed by executing HM,TS,DM<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - No
Contour Mode - No

Example:

```
;     This routine makes a 12000 count move,
;     and reports actual position when the servo
;     is within 10 counts of the target position.

DB10,+        ;Set target deadband to 10
MR12000,+     ;Start move
WT0,+         ;Wait for servo in range
TP+           ;Report position.
```

Related Commands: **WT,DT**

DD*n*

Direction deadband

This command can be used to minimize servo dithering when the DSC is positioning a low friction load.. The value ***n*** sets the number of servo update periods over which a torque direction change is delayed and torque magnitude is held at zero.

See "Suggestions for Servo Loop Tuning" in this manual for additional information.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - Yes
Contour Mode - Yes

Range: 0 <= ***n*** <= 255

Registers: Servo Table, offset 111

When the DD command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 111, decimal. The value ***n*** can be accessed by executing RB111,TR0<*CR*>. The default value is 0.

Related Commands: **OD, OO**

DG*n*

Deceleration gain

This command sets the deceleration feed forward gain for the servo. This feed forward produces a torque value proportional to the set deceleration rate, when the servo is decelerating. This torque value is numerically summed with the output of the PID algorithm.

Once this value is set, deceleration rates may be changed, and the corresponding feed forward value will be determined automatically.

See "Suggestions for Servo Loop Tuning" in this manual for additional information.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - No
Contour Mode - No

Example:

```
;       This routine causes the deceleration rate to
;       change while the servo is in motion.

VM,DG100,+      ;Vel. mode, set dec. Gain.
SV4000000,+     ;Set target velocity.
DS1000,GO,+     ;Set decel, start move.
WA10000,+       ;Wait for 5 seconds.
ST,+            ;Start deceleration to stop.
WA3000,+        ;Wait for 1.5 seconds.
DS50+           ;Change decel rate to slow
```

Range: -32,767 <= ***n*** <= 32,767

Registers: Servo Table, offset 80

When the DG command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 80, decimal. The value ***n*** can be accessed by executing RW80,TR0<*CR*>. The default value is 0.

Related Commands: **AG, SA, DS**

DS*n*

Deceleration Set

This command sets the deceleration rate for the servo. The 32 bit parameter for this command is interpreted as having a 16 bit integer part and a 16 bit fractional part. This number determines how much the servo's velocity will be decreased by each 1/2 millisecond while decelerating.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - No
Contour Mode - No

Range: 0 <= ***n*** < 2,147,483,647

Registers: Servo Table, offset 46

When the DS command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 46, decimal. The value ***n*** can be accessed by executing RL46,TR0<*CR*>. The default value is 5000.

Example:

; This routine performs a trapezoidal move.

```
DH,+          ;Define home pos.
SA5000,+      ;Set acceleration rate.
DS5000,+      ;Set deceleration rate.
SV1000000,+   ;Set maximum velocity.
MA40000+      ;Move to absolute pos.
```

Related Commands: **DG, SA, AG**

FR*n*

Set derivative sample period

This command helps tune the servo loop to the inertial characteristics of the load. High inertial loads normally require a longer period, and low inertial loads a shorter period. For a value ***n***, the sample period will be (***n***+1)*0.5 milliseconds.

See "Suggestions for Servo Loop Tuning" in this manual for additional information.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - Yes
Contour Mode - Yes

Range: 0 <= ***n*** <= 127

Registers: Servo Table, offset 110

When the FR command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 110, decimal. The value ***n*** can be accessed by executing RB110,TR0<*CR*>. The default value is zero.

Example:

; This routine initializes the PID loop.

```
WS100,+       ;Wait for Stop.
MF,+          ;Turn motor off.
SG200,+       ;Set P gain.
SI5,+         ;Set I gain.
IL250,+       ;Set I limit
SD200,+       ;Set D gain.
FR0,+         ;Set D period.
MN+           ;Turn motor on.
```

Related Commands: **SD**

IL*n*

Set integration limit

This command sets a maximum limit for the integration of position error. The percentage of maximum motor torque that can be produced by the integrator term alone, can be calculated as follows:

"% of Max Torque" = (IL***n***)*(SI***n***) / 2620
(where Max Torque = Motor Kt * DSC peak output current rating.)

See "Suggestions for Servo Loop Tuning" for additional information.

Range: 0 < ***n*** < 32,767

Registers: Servo Table, offset 94

When the IL command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 94, decimal. The value ***n*** can be accessed by executing RW94,TR0<*CR*>. The default value is 50.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - Yes
Contour Mode - Yes

Example:

; This routine initializes the PID loop.

```
WS100,+      ;Wait for Stop.
MF,+         ;Turn motor off.
SG200,+      ;Set P gain.
SI5,+        ;Set I gain.
IL250,+      ;Set I limit.
SD200,+      ;Set D gain.
FR0,+        ;Set D period.
MN+          ;Turn motor on.
```

Related Commands: **IM, SI, SQ**

IM*n*

Integration mode

This command enables or disables the position error integrator during a move. If IM0 is executed, the DSC will only enable the integrator function after the trajectory generator has reached a target position. If IM1 is executed, the DSC will keep the integrator enabled at all times. Upon reset, the DSC initializes to IM0.

See "Suggestions for Servo Loop Tuning" in this manual for additional information.

Range: ***n*** = 0: Disable I during move.
= 1: Enable I during move.

Registers: Servo Status, bit 12

Usage:
Position Mode - Yes
Velocity Mode - Yes
(Note 1) Gain Mode - No
Torque Mode - No
(Note 1) Slave Mode - No
(Note 1) Contour Mode - No

Note 1. The integrator is always enabled in Gain, Slave and Contour modes.

Related Commands: **IL, SI, SQ**

LF*n*

Limit switch off

Disables one or more limit switch inputs. The parameter to this command determines which inputs will be disabled. The coding of the parameter is the same as for the limit switch on command, LN.

Range: ***n*** = none: limit+ and limit-
= 0: limit+ and limit-
= 1: limit+
= 2: limit-

Registers: Servo Status, bits 27 and 30.

When the LF command is executed, bits 27 and 30 are set to 0. When set, bit 27 is limit- enabled, and bit 30 is limit+ enabled. Individual bits may be set to zero by using 1 or 2 for the parameter *n*. The default value is 0 for both bits.

Usage: Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This routine locates, then reports the
;     approximate position of the positive limit
;     input.

LN1,LM2+          ;Lim+ on w/ smooth stop.
VM,SV10000,+      ;Vel. mode & slow  speed.
DI0,GO,+          ;Set direction + & move.
WS,LF1+           ;Wait for stop., Lim+ off.
MN,TP,+           ;Enable servo & tell pos.
PM,GH+            ;Position mode, go home.
```

Related Commands: **LN, LM**

LM*n*

Limit switch mode

This command is used to select how the DSC will react when a limit switch is detected. As shown in the table below, "Coast" will cause the servo to be disabled, "Abrupt" will freeze the trajectory generator, but leave the servo enabled, and "Smooth" will cause the trajectory generator to decelerate the motor to a stop.

The DSC will ignore limit inputs unless they are enabled via the limit on command, LN, and the servo is in motion.

Range: ***n*** = 0: Coast to stop
= 1: Abrupt stop
= 2: Smooth stop

Registers: Servo Status, bits 16, 24 & 25

In all limit modes, when a limit input is enabled and detected, the error flag in the servo status register, bit 16, will be set. This will prevent the DSC from moving the servo. A motor on command, MN is used to clear bit 16.

Bits 24 and 25 encode the selected limit mode, *n*, above.

Usage: Position Mode - Yes
Velocity Mode - Yes
Gain Mode - No
(Note 1) Torque Mode - Yes
(Note 2) Slave Mode - Yes
(Note 2) Contour Mode - Yes

Note 1. In Torque Mode, the trajectory generator and PID loop are not active. LM1 and LM2 will have no effect. LM0 can be used. Limit inputs are detected independent of motor direction.

Note 2. In Contour and Slave Mode, the trajectory generator is not active. LM2 will have no effect. LM0 and LM1 can be used. Limit inputs are detected independent of motor direction.

Example: See routines given for LN and LF.

Related Commands: **LN, LF**

LN*n*

Limit switch on

Enables one or more limit switch inputs. The parameter to this command determines which inputs will be disabled. The coding of the parameter is the same as for the limit switch off command, LF.

Range: ***n*** = none: limit+ and limit-
= 0: limit+ and limit-
= 1: limit+
= 2: limit-

Registers: Servo Status, bits 27 and 30.

When the LN command is executed, bits 27 and 30 are set to 1. When set, bit 27 is limit- enabled, and bit 30 is limit+ enabled. The default value is 0 for both bits.

Usage: Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;   This routine locates, then reports the
;   approximate position of the positive limit
;   input.

LN1,LM2+        ;Lim+ on & smooth stop.
VM,SV10000,+    ;Vel. mode & slow  speed.
DI0,GO,+        ;Set direction + & move.
WS,LF1+         ;Wait for stop., Lim+ off.
MN,TP,+         ;Enable servo & tell pos.
PM,GH+          ;Position mode, go home.
```

Related Commands: **LF, LM**

OD*n*

Set output deadband

This command operates on the position PID loop, torque output value. The value *n* is added to the torque output, if the torque output is greater than zero, or subtracted from the torque output, if the torque output is less than zero. This command creates a "jump around zero torque" function which can be used to eliminate the effects of frictional deadband within a given system.

See "Suggestions for servo loop tuning" in this manual for additional information.

Range: 0 <= ***n*** < = 1023

Registers: Servo Table, offset 106

When the OD command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 106, decimal. The value ***n*** can be accessed by executing RW106,TR0<*CR*>. The OD default value is 0.

Usage: Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;   This routine initializes output torque limit,
;   deadband, and offset values.

MF,PM,+         ;Motor off & pos. mode.
SQ818,+         ;Set torque limit to 80%.
OD51,+          ;Set deadband to 5%.
OO102,+         ;Set torque offset to 10%.
MN+             ;Motor on.
```

Related Commands: **SQ, OO**

OO*n*

Set output offset

This command operates on the position PID loop, torque output value. The value ***n*** is added to the torque output value after operated on by the OD function. This feature can be used to compensate for a gravitational load by setting a value ***n*** equal to the amount of torque required to hold position against gravity. Use of the OO command can eliminate dependence on the position loop integrator.

See "Suggestions for servo loop tuning" in this manual for additional information.

Range: -1023 <= ***n*** < = 1023

Registers: Servo Table, offset 100

When the OO command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 100, decimal. The value ***n*** can be accessed by executing RW100,TR0<*CR*>. The OO default value is 0.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;   This routine initializes output torque limit,
;   deadband, and offset values.

MF,PM,+     ;Motor off & pos. mode.
SQ818,+     ;Set torque limit to 80%.
OD51,+      ;Set deadband to 5%.
OO102,+     ;Set torque offset to 10%.
MN+         ;Motor on.
```

Related Commands: **SQ, OD**

PH*n*

Set servo output phase

This command is used to set the servo's feedback phasing. The phase of the feedback will determine whether the DSC drives the servo in a direction that reduces, or increases position error. This command is functionally equivalent to swapping the wiring of encoder channels A and B.

See "First time servo operation" in this manual for more information.

Range: ***n*** = 0: Standard phase
= 1: Reverse phase

Registers: Servo Status, bit 8

When the PH command is executed, bit 8 in the servo status register is set or cleared as determined by the value ***n***. The default setting is PH0.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - Yes
Contour Mode - Yes

Related Commands: **none**

SA*n*

Set acceleration

This command sets the acceleration rate for the servo. The 32 bit parameter to this command is interpreted as having a 16 bit integer part and a 16 bit fractional part. This number determines how much the servo's velocity will be increased by each 1/2 millisecond during acceleration.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - No
Contour Mode - No

Range: 0 <= ***n*** < 2,147,483,647

Registers: Servo Table, offset 42

When the SA command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 42, decimal. The value ***n*** can be accessed by executing RL42,TR0<*CR*>. The default value is 5000.

Example:

```
;   This routine causes the acceleration rate to
;   change while the servo is in motion.

DH,+          ;Define as home.
AG250,+       ;Set accel gain to 250.
SA500,+       ;Set accel to low value.
MR40000,+     ;Start move.
WP4000,+      ;Wait for position 4000.
SA25000+      ;Now set accel to high.
```

Related Commands: **AG, DS, DG**

SD*n*

Set derivative gain

This command helps stabilize servo response. This command can also reduce lag associated with rapidly changing trajectory or load conditions. In general, higher values of *n* will produce tighter dynamic response, and lower values will produce looser dynamic response. The value ***n*** may be modified when the servo is in motion or at rest.

See "Suggestions for servo loop tuning" in this manual for additional information.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - Yes
Contour Mode - Yes

Range: 0 <= ***n*** < = 32,767

Registers: Servo Table, offset 72

When the SD command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 72, decimal. The value ***n*** can be accessed by executing RW72,TR0<*CR*>. This value should be initialized after power is applied to the DSC and before servo motion is commanded. The default value is 200.

Example:

```
;   This routine changes the derivative gain
;   after a move has been completed.

PM,+          ;Switch to Position Mode.
SD4000,+      ;Set high D for move.
MR40000,+     ;Start move.
WS200,+       ;Wait 100 mS after move.
SD200+        ;Set low D for pos. hold.
```

Related Commands: **FR, SG, SI**

SE*n*

Set following error

Used to set the maximum permissible following error for the servo. Following error is the encoder count difference between the servo's actual position and the servo's commanded position. When excessive following error is detected, the servo will be turned off, an "F." will appear on the DSC status display, and the motor will coast to a stop. A MF, or MN command can be used to recover from this fault condition.

Range: 0 <= ***n*** < = 32,767

Registers: Servo Table, offset 98
Servo Status, bits 16 and 17

When the SE command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 98, decimal. The value ***n*** can be accessed by executing RW98,TR0<*CR*>. When following error is detected, bit 16, motor error, and 17, following error, in the servo status register will be set to 1. A MN command will clear these bits.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine reports the actual level of
;       following error while the servo is moving.

SE1000,+        ;Set max following error.
VM,+            ;Change to velocity mode.
GO,+            ;Begin move.
WA500,+         ;Wait for 250 mS.
TF+             ;Tell following error.
```

Related Commands: **TF**

SF*n*

Slave input format

This command initializes the DSC for slave mode operation. The parameter ***n*** encodes the type of auxiliary feedback and operational features that are functional in slave mode. This parameter should be initialized prior to execution of the slave mode command, SM.

Range:	Feedback	Phasing	IN4 sync.
n = 0	quad. enc.	standard	off
n = 1	step+dir.	standard	off
n = 2	quad. enc.	reverse	off
n = 3	step+dir.	reverse	off
n = 4	quad. enc.	standard	on
n = 5	step+dir.	standard	on
n = 6	quad. enc.	reverse	on
n = 7	step+dir.	reverse	on

Registers: Servo Table, offset 113

The value ***n*** can be accessed by executing RB113,TR0<*CR*>. The default value for ***n*** is zero.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - No
Contour Mode - Yes

Example:

```
;       This routine initializes the DSC and then
;       transitions to slave mode.

SF0,+           ;Quadrature, standard, off.
SO200,+         ;Slave offset is 200 counts.
DI0,+           ;Slave direction is 0.
RL54,+          ;Put slave pos. in reg. 0
DH@0,+          ;Init. servo to slave pos.
SM268435456+    ;Start slave with ratio 1:1.
```

Related Commands: **DI, SM, SO**

SG*n*

Set proportional gain

This command sets the level of proportional gain in the position PID loop. This command has a "spring" like effect on the servo's response to position error. High values of ***n*** will create stiffness and tight response to position error. Low values of ***n*** will create a loose response position error.

See "Suggestions for servo loop tuning" in this manual for additional information.

Range: 0 <= ***n*** < = 32,767

Registers: Servo Table, offset 70

When the SG command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 70, decimal. The value ***n*** can be accessed by executing RW70,TR0<*CR*>. The default value is 200.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;    This routine changes the proportional gain
;    after a move has been completed.

PM,+          ;Switch to Position Mode.
SG500,+       ;Set low P for move.
MR40000,+     ;Start move.
WS200,+       ;Wait 100 mS after move.
SG1000+       ;Set high P when holding.
```

Related Commands: **SD, SI**

SI*n*

Set integral gain

This command sets the integral gain level for the position PID loop. This integrator operates on the persistence of position error. Higher values of ***n*** will cause any persisting error to be eliminated more rapidly, but may also cause servo instability. The effect of the SI command is also limited by the IL command.

See "Suggestions for servo loop tuning" in this manual for additional information.

Range: 0 <= ***n*** < = 32,767

Registers: Servo Table, offset 74
Servo Status, bit 12

When the SI command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 74, decimal. The value ***n*** can be accessed by executing RW74,TR0<*CR*>. Bit 12 in the servo status register, integration mode, has a default value of 0. See the IM command for additional details.

Usage:
(Note 1) Position Mode - Yes
(Note 1) Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - Yes
Contour Mode - Yes

Note 1. In Position or Velocity mode, the error integrator will only be active when the servo is on, and holding position. IM1 must be executed to enable integration of position error while the servo is moving.

Example:

```
;    This routine enables the error integrator
;    during motion.

IM1,SI30,+    ;Set Integral mode & gain.
VM,GO+        ;Vel Mode and start move.
```

Related Commands: **IM, IL, SG, SD**

SO*n*

Slave offset

This command initializes the DSC for slave mode operation. The parameter ***n*** sets an encoder count offset between the slave input position and the servo's commanded position. This parameter should be initialized prior to execution of the slave mode command, SM.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - No
Contour Mode - Yes

Example:

```
;       This routine initializes the DSC and then
;       transitions to slave mode.

SF0,+           ;Quadrature, standard, off.
SO200,+         ;Slave offset is 200 counts.
DI0,+           ;Slave direction is 0.
RL54,+          ;Put slave pos. in reg. 0
DH@0,+          ;Init. servo to slave pos.
SM268435456+    ;Start slave with ratio 1:1.
```

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: Servo Table, offset 12

When the SO command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 12, decimal. The value ***n*** can be accessed by executing RL12,TR0<*CR*>. The default value is zero.

Related Commands: **DI, SF, SM**

SQ*n*

Set torque

This command sets an absolute limit on the torque level that is produced by the position PID loop. The value *n* / 10 equals the percentage of maximum torque output for a given DSC package. To find the maximum package torque, multiply the DSC's peak current rating by the motor's torque constant, Kt.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
(Note 1) Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Note 1. In Torque Mode, the position PID loop is disabled. In this mode, the SQ command will set torque levels directly.

Example:

```
;       This routine transitions the DSC from
;       any mode to Torque Mode.

MF,+            ;Turn motor off.
SQ0,+           ;Make sure torque is zero.
QM,+            ;Switch to Torque Mode.
MN+             ;Then turn motor on.
```

Range: 0 <= ***n*** < = 1,023

Registers: Servo Table, offset 96

When the SQ command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 96, decimal. The value ***n*** can be accessed by executing RW96,TR0<*CR*>. The default value is 1023, or 100%.

Related Commands: **IL, QM**

SV*n*

Set velocity

This command sets maximum servo velocity in units of quadrature counts per servo update period. The 32 bit parameter ***n*** is interpreted as having a 16 bit integer part and a 16 bit fractional part.

Example: Assuming a 1000 line encoder determine *n* for a limit of 1500 rpm.

1500 rpm / 60 sec/min = 25 rps
25 rps * 4000 quad. cnts/rev. = 100000 cps
100000 cps / 2000 updates/s = 50 cnts/update
(Now shift decimal point 16 bits)
50 * 65536 = 3276800

n = 3,276,800

Range: 0 <= ***n*** < 2,147,483,647

Registers: Servo Table, offset 38

This value ***n*** can be accessed by executing RL38,TR0<*CR*>. The default value is 1,000,000.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - No
(Note 1) Contour Mode - Yes

Note 1. In Contour Mode, the SV command sets a time value for traveling between points. This time is calculated as (65536 / *n*) * 0.0005 seconds. For uniform motion, only values of ***n*** that are powers of 2 should be used. See "Contour Mode operation" for additional information.

Example:

```
;       This routine sets the maximum velocity

PM+             ;Set DSC to pos. mode.
SV3276800+      ;Set Velocity to 1500 rpm.
```

Related Commands: **VG, SA, DS**

VG*n*

Velocity Gain

This command sets velocity feed forward gain for the servo. This function produces a torque value proportional to the velocity command from the trajectory generator, when the servo is in motion. This torque value is numerically summed with the output of the PID algorithm. Once ***n*** is set, velocities rates may be changed, and the corresponding torque value will be determined automatically.

See "Suggestions for Servo Loop Tuning" in this manual for additional information.

Range: -32,767 <= ***n*** < = 32,767

Registers: Servo Table, offset 76

When the VG command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 76, decimal. The value ***n*** can be accessed by executing RW76,TR0<*CR*>. The default value is zero.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - No
Contour Mode - No

Example:

```
;       This routine causes velocity to change when
;       the servo is in motion.

SV10000,+       ;Set low velocity.
VG100,+         ;Set velocity feed forward.
MR40000,+       ;Start move.
WR-36000,+      ;Wait for 4000 counts.
SV1000000,+     ;Set high velocity.
WS50+           ;Wait for stop plus 25 mS.
```

Related Commands: **SV, AG, DG**

AB

Abort Motion

This command stops the servo abruptly, but leaves the servo and position PID loop enabled. When AB is executed, the trajectory generator function is aborted by setting the target position equal to the present position and holding it at this value.

The following error limit, as set by the SE command, may be exceeded when the AB is executed. If this occurs, the servo will be disabled, an "F." will appear on the DSC status display, and the motor will coast to a stop. The SE value must be set to a sufficiently large number for the AB command to operate.

Registers: Servo Table, offsets 20 and 24

Offsets 20 and 24 are system registers for storage of "target" and "trajectory" positions respectively. Target is the final destination for a move, and trajectory is the present commanded servo position. Either value may be evaluated by executing a RL<offset>,TR0<*CR*>.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - No
Contour Mode - No

Example:

```
;   This routine aborts motion 100 mS after a
;   move is started.

PM,+            ;Change to Position Mode.
SE4000,+        ;Set large following error.
SV1000000,+     ;Set max velocity level.
MR400000,+      ;Start move.
WA200,+         ;Wait for 100 mS.
AB+             ;Abort move.
```

Related Commands: **ST, MF, SE**

CM*n*

Contour Mode

This command sets the DSC to Contour mode operation The value ***n*** is the starting index number to servo point memory. In contour mode, the DSC creates trajectories by successively reading positions from servo point memory, and linearly interpolating between these points.

Learn target, LT, and learn point, LP, are used to fill servo point memory. Also, MA, MR, SV, TT and TX commands offer unique operation specific to contour mode. See "Trajectory generation modes" for additional information.

Range: 0 <= ***n*** <= 2047

Registers: Servo status, bit 13
Servo point memory

When the CM command is executed, bit 13 of the servo status register is set to 1. Other mode bits: 9, 10, 11 and 14 are set to 0.

When motion is commanded in contour mode, trajectory points are read by index number from servo point memory, 0 to 2047.

Examples:

```
;   This routine programs points 0 to 4 in
;   servo point memory.

PM,MF,RM2,+         ;Pos Mode, reset points.
DH,LT0,+            ;Def. home, learn point 0.
MR10000,LT1,+       ;Set and learn point 1.
MR-5000,LT2,+       ;Set and learn point 2.
MR,5000,LT3,+       ;Set and learn point 3.
MR-10000,LT4+       ;Set and learn point 4.

;   This routine moves from point 0 to 4 in
;   servo point memory.

DH,MN,+             ;Define home, motor on.
CM0,SV32,+          ;Start point & travel time.
MA4+                ;Move from point 0 to 4.
```

Related Commands: **MA, MR, SV, LP, LT**

DH*n*

Define home

This command defines the present servo position as the home position and equal to the value *n*. After DH is executed, a move absolute to zero command, MA0, will position the servo to ***-n*** counts from home. A go home command, GH, will position the servo to ***n***.

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: Servo Table, offset 4

When the DH command is executed, the servo's measured position is set to zero. Absolute position, as operated on by TP and GH, are calculated as: Absolute position = servo's measured position + *n*. The servo's measured position may be evaluated by executing RL4,TR0<*CR*>.

Usage:

Position Mode - Yes
Velocity Mode - No
Gain Mode - Yes
Torque Mode - No
Slave Mode - No
Contour Mode - No

Example:

```
;    This routine defines home as +1000 counts
;    from the servo's encoder index mark.

PM,SV100000,+      ;Pos Mode & slow  speed.
MR10000,+          ;Start multiple rev. move.
FI,+               ;Find index when moving.
WS100,+            ;Wait for move to end.
MA1000,+           ;Move to index + 1000.
WS100,+            ;Wait for move to end.
DH1000+            ;Define as home position.
```

Related Commands: **MA, GH, FI**

DI*n*

Direction

This command sets the servo's direction for either velocity of slave mode operation.

In velocity mode, the DSC considers positive direction to be when the servo's encoder count is increasing, and negative direction to be when the servo's encoder count is decreasing. In slave mode, positive direction is when the servo's encoder count increases as the auxiliary input count increases, and negative, when the servo's encoder count decreases as the auxiliary count increases.

This command should <u>not</u> be confused with the servo phasing command, PH, which determines whether the position PID loop in closed with negative of positive feedback.

Range:

n	=	0:	Positive
	=	1:	Negative

Registers: none.

The default value for direction is 0.

Usage:

Position Mode - No
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - Yes
Contour Mode - No

Examples:

```
;    This routine uses the DI command to
;    when the servo is moving in velocity mode.

VM,SV1000000,+     ;Set Vel mode and speed.
GO,WA2000,+        ;Start move & wait 1 sec..
DI1+               ;Set to negative direction.
```

Related Commands: **VM, SM**

FI*n*

Find index

This command is used to find the servo's index position. FI suspends operation of the command interpreter until the servo encoder index pulse is detected. For this reason, servo motion must be initiated before FI is executed.

When the index pulse is detected, the present position of the servo is captured, and the index position is defined as: (*captured pos.* - ***n***). The index position is then stored, and command interpretation is continued. The initiating motion preceding FI continues to the original target until completed, stopped, or aborted by execution of a new motion command.

After the index pulse is detected, TP and MA commands calculate absolute position as: (*present pos - index pos*).

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: Servo Table, offsets 4 and 8

Present and index positions are at offsets 4 and 8 respectively. To evaluate: RL<offset>,TR0.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - No
Contour Mode - No

Example:

```
;    This routine defines home as +1000 counts
;    from the servo's encoder index mark.

PM,SV100000,+      ;Pos Mode & slow speed.
MR10000,+          ;Start multiple rev. move.
FI0,+              ;Find index when moving.
WS100,+            ;Wait for move to end.
MA1000,+           ;Move to index + 1000.
WS100,+            ;Wait for move to end.
DH1000+            ;Define as home position.
```

Related Commands: **DH**

GH

Go home

This command will move the servo to the home position, as defined by the DH command. If in position mode, the move will be controlled by the trajectory generator. If in gain mode, the move will be abrupt.

Range: none.

Registers: Servo status, bits 1 and 2.

After execution of the GH command, *trajectory complete*, bit 1, and *at target*, bit 2, will be set to 0 in the servo status register. When, the trajectory generator has indexed to the target position, bit 1 will be set to 1. Bit 2 will be set to 1 after bit 1, and when the servo has reached the target deadband as defined by the DB command.

Usage:
Position Mode - Yes
Velocity Mode - No
Gain Mode - Yes
Torque Mode - No
Slave Mode - No
Contour Mode - No

Example:

```
;    This routine creates motion in velocity mode,
;    transitions to position mode, then moves the
;    servo back to the home position.

VM,SV1000000,+     ;Set Vel mode and speed.
GO,WA2000,+        ;Start move & wait 1 sec.
ST,WS100,+         ;Stop & wait for complete.
PM,GH+             ;Set Pos mode & go home.
```

Related Commands: **DH, MA, MR**

GM

Gain mode

This command places the DSC in gain mode operation. In this mode, the DSC executes moves commands, however, no trajectory profile is generated. When a move command is executed in gain mode, the trajectory generator position is immediatly updated to the given target position, and the servo is driven to this target by the output of the position PID loop alone.

This mode of operation can be used to evaluate the position step response and stability of the servo system. See "Suggestions for servo loop tuning" in this manual for additional information.

Range: none

Registers: Servo Status, bits 9 - 11, 13, 14

When the GM command is executed, usage mode bits; 9 to 11, 13 and 14 are set to zero.

Usage: See "Trajectory generation modes" in this manual for additional information.

Example:

```
;     This routine creates motion in gain mode,
;     transitions to position mode, then moves the
;     servo back to the home position.

GM,+              ;Set DSC to Gain mode.
MR500,+           ;Start short move.
WS10000,+         ;Wait for complete +5 sec.
PM,GH+            ;Set Pos mode & go home.
```

Related Commands: PM, VM, SM, CM, QM

GO

Go to velocity

The GO command is used to initiate motion in velocity mode. When GO is executed, the trajectory generator will accelerate the servo in the direction set by the DI command, at the rate set by the SA command, to the velocity set by the SV command.

Range: none.

Registers: Servo status, bits 1 and 2.

After execution of the GO command, servo status bit 1, *trajectory complete*, and bit 2, *at target*, will be reset to 0. When a ST command is executed, the trajectory generator will index to a target position , and bit 1 will be set when complete. Bit 2 will be set after bit 1, and when the servo has reached the target deadband, as defined by the DB command.

Usage:

Position Mode - No
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - No
Contour Mode - No

Examples:

```
;     This routine creates motion in velocity mode,
;     transitions to position mode, then moves the
;     servo back to the home position.

VM,SV1000000,+    ;Set Vel mode and speed.
GO,WA2000,+       ;Start move & wait 1 sec..
ST,WS100,+        ;Stop & wait for complete.
PM,GH+            ;Set Pos mode & go home.
```

Related Commands: ST, SV, DI, SA, DS

MA*n*

Move absolute

This command generates a motion to the absolute position ***n***. Absolute position is relative to the point of initialization as set by DH or FI. The servo must be enabled by MN for motion to occur. The servo target position will be adjusted whether the servo is enabled or not.

Motion created by MA is depended on the selected trajectory mode. See "Trajectory mode operation" for additional information.

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: Servo Status, bits 1 and 2.
Servo Table, offset 20.

After execution of MA, servo status bit 1, *trajectory complete*, and bit 2, *at target*, will be reset. When the trajectory generator has indexed to the target position , bit 1 will be set. Bit 2 will be set after bit 1, and when the servo has reached the target deadband, as defined by DB. The *move target* value is stored at offset 20 decimal. in the servo table.

Usage:
Position Mode - Yes
Velocity Mode - No
Gain Mode - Yes
Torque Mode - No
Slave Mode - No
(Note 1) Contour Mode - Yes

Note 1. In Contour Mode, the MA value ***n*** is specified as an index number to servo point memory. In this mode, the range of *n* is from 0 to 2047.

Example:

```
;       This routine performs a trapezoidal move.

DH,+                ;Define home pos.
SA5000,DS5000,+     ;Set accel & decel rate.
SV1000000,+         ;Set maximum velocity.
MA40000+            ;Move to absolute pos.
```

Related Commands: **DH, FI, GH, MR**

MF

Motor off

This command is used to disable the servo amplifier internal to the DSC. When disabled, the DSC fault relay output will open and the motor will coast to a stop.

When the servo is disabled, the DSC will continue to execute commands but no motion will occur.

Warning!: The MF command should not be used as part of a safety interlock system. This command only provides a solid state method for disabling the servo, and as such, MF is not fail safe.

Range: none.

Registers: Servo Status, bit 0.

After execution of the MF command, servo status bit 0, *motor on*, will be reset to 0, indicating that the motor is disabled.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine transitions the DSC from
;       any mode to Torque Mode.

MF,+            ;Turn motor off.
SQ0,+           ;Make sure torque is zero.
QM,+            ;Switch to Torque Mode.
MN+             ;Then turn motor on.
```

Related Commands: **MN**

MN

Motor on

This command is used to enable the servo amplifier internal to the DSC. When MN is executed with position, velocity, gain, or contour mode selected, the DSC will reset the fault relay output, and hold the motor at its present position. When MN is executed in torque mode or slave mode, motion may occur.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine transitions the DSC from
;       any mode to Torque Mode.

MF,+            ;Turn motor off.
SQ0,+           ;Make sure torque is zero.
QM,+            ;Switch to Torque Mode.
MN+             ;Then turn motor on.
```

Range: none.

Registers: Servo Status, bit 0.

After execution of the MN command, servo status bit 0, *motor on*, will be set to 1, indicating that the motor is enabled.

Related Commands: **MF**

MR*n*

Move relative

This command generates a motion to the relative position ***n***. Relative position is the servo's present position plus ***n*** encoder counts. The servo must be enabled by MN for motion to occur. The servo target position will be adjusted whether the servo is enabled or not.

Motion created by MR is depended on the selected trajectory mode. See "Trajectory mode operation" for additional information.

Usage:
Position Mode - Yes
Velocity Mode - No
Gain Mode - Yes
Torque Mode - No
Slave Mode - No
(Note 1) Contour Mode - Yes

Note 1. In Contour Mode, the MR value ***n*** is specified as an index number to servo point memory. In this mode, the range of *n* is from 0 to 2047.

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: Servo Status, bits 1 and 2.
Servo Table, offset 20.

After execution of MR, servo status bit 1, *trajectory complete*, and bit 2, *at target*, will be reset. When the trajectory generator has indexed to the target position , bit 1 will be set. Bit 2 will be set after bit 1, and when the servo has reached the target deadband, as defined by DB. The *move target* value is stored at offset 20 decimal. in the servo table.

Example:

```
;       This routine performs a trapezoidal move.

DH,+                ;Define home pos.
SA5000,DS5000,+     ;Set accel & decel rate.
SV1000000,+         ;Set maximum velocity.
MR40000             ;Move to absolute pos.
```

Related Commands: **MA**

PM

Position mode

This command places the DSC into position mode. In this mode, the DSC executes move commands based on an internally generated, trapezoidal trajectory profile. During a move, the trajectory generator incrementally advances commanded position, as determined by SA , DS and SV settings, until the target position is reached. Feed forward terms, AG, DG, and VG are also active in this mode of operation.

Range: none

Registers: Servo Status, bit 9

When the PM command is executed, bit 9 of the servo status register is set and bits 10, 11, 13 and 14 are reset..

Usage: See "Trajectory generation modes" in this manual for additional information.

Example:

```
;       This routine creates motion in position
;       mode.

PM,+            ;Set DSC to pos. mode.
SA5000,+        ;Set acceleration rate.
DS5000,+        ;Set deceleration rate.
SV100000,+      ;Set velocity rate
MR12000,+       ;Move relative 12000.
WS1000,+        ;Wait complete + 0.5sec.
TP+             ;Tell position.
```

Related Commands: GM, VM, SM, CM, QM

QM

Torque mode

This command places the DSC in torque mode. In this mode, the internal trajectory generator is disabled, the position PID loop is turned off, and move commands are ignored. The torque limit command, SQ, is used to set commanded torque (current) level directly.

Caution should be taken when transitioning the DSC to torque mode operation. When QM is executed, the last internally generated torque command will be held. The DSC will make to attempt to limit motor speed, or fault on following error.

Range: none

Registers: Servo Status, bit 11

When the QM command is executed, bit 11 of the servo status register is set, and bits 9, 10, 13 and 14 are reset..

Usage: See "Trajectory generation modes" in this manual for additional information.

Example:

```
;       This routine transitions the DSC from
;       any mode to Torque Mode.

MF,+            ;Turn motor off.
SQ0,+           ;Make sure torque is zero.
QM,+            ;Switch to Torque Mode.
MN+             ;Then turn motor on.
```

Related Commands: GM, VM, SM, CM, PM

SM*n*

Slave mode

This command places the DSC into slave mode. In this mode, the DSC positions the servo based on the ratio ***n***, and an externally generated, slave input position. The ratio ***n*** is a real value between 0 and 16, and is represented as a 32 bit number with an implied decimal point between bit 27 and 28. Related commands; slave format (SF), slave offset (SO) and direction (DI) are used to initialize the DSC prior to execution of SM. The following formula is used to determine servo position after execution of SM:

Servo pos.= (present slave input pos. - initial slave input pos.)* ratio*dir.+ offset.

Range: 0 <= ***n*** < 2,147,483,647

Registers: Servo Status, bit 14
Servo Table, offset 62

When the SM command is executed, bit 14 of the servo status register is set and bits 9-11 and 13 are reset. The value *n* can be read by executing RL62,TR0<*CR*>.

Usage: See "Trajectory generation modes" in this manual for additional information.

Example:

```
;       This routine initializes the DSC and then
;       transitions to slave mode.

SF0,+           ;Slave input is quadrature.
SO200,+         ;Slave offset is 200 counts.
DI0,+           ;Slave dirrection is 0.
RL54,+          ;Put slave postion in reg. 0
DH@0,+          ;Init. servo to slave pos.
SM268435456+    ;Slave mode, ratio = 1:1.
```

Related Commands: **SF, SO, DI**

ST

Stop

The ST command is used to end motion in velocity mode. When ST is executed, the trajectory generator will decelerate the servo in the direction set by the DI command, at the rate set by the DS command, to zero velocity, and then hold postion.

Range: none.

Registers: Servo status, bits 1 and 2.

When the ST command is executed, the trajectory generator will index to a target position as determined by the DS command, and bit 1 of the servo status register will be set when complete. Bit 2 will be set after bit 1, and when the servo has reached the target deadband, as defined by the DB command.

Usage:
Position Mode - No
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - No
Contour Mode - No

Example:

```
;       This routine creates motion in velocity
;       mode, transitions to position mode, then
;       moves the servo back to the home position.

VM,SV1000000,+  ;Set Vel mode and speed.
GO,WA2000,+     ;Start move & wait 1 sec..
ST,WS100,+      ;Stop & wait for complete.
PM,GH+          ;Set Pos mode & go home.
```

Related Commands: **GO, SV, DI, SA, DS**

VM

Velocity mode

This command places the DSC in velocity mode operation. In this mode, the DSC creates velocity by allowing the trajectory generator to continuously advance commanded position at a rate set by SV. When a new velocity set point is issued, the servo will accelerate or decelerate as determined by SA and DS values respectively. GO and ST commands are used to initiate and terminate motion in velocity mode.

Range: none

Registers: Servo Status, bit 10

When the VM command is executed, bit 10 of the servo status register is set and bits; 9, 11, 13 & 14 are reset.

Usage: See "Trajectory generation modes" in this manual for additional information.

Examples:

```
;   This routine creates motion in velocity mode,
;   transitions to position mode, then moves the
;   servo back to the home position.

VM,SV1000000,+      ;Set Vel mode and speed.
GO,WA2000,+         ;Start move & wait 1 sec..
ST,WS100,+          ;Stop & wait for complete.
PM,GH+              ;Set Pos mode & go home.
```

Related Commands: PM, GM, SM, CM, QM

AT

Tell auxiliary position

This command reports the present position of the auxiliary slave input in units of step or quadrature counts as determined by the slave format command, SF.

The reported value is limited to +/- 2,147,483,647 decimal. The value is formatted with leading zeros and a minus character if negative.

Range: none

Registers: Servo Table, offset 54.

RL54<*CR*> will load this value into the DSC's accumulator, register 0.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This is a command line example where AT
;     is entered at the DSC prompt.

>AT              ;<CR>
-0000008233      ;A value +<CR> is returned,
>                ;and then a new prompt.
```

Related Commands: **SM, SF, SO, DI**

DO*n*

Display optimal

This command reports the optimal position of the servo at time ***n***, relative to initiation of position record function, PR. The parameter ***n*** is an index to recorded data and is in units of multiple servo update periods as determined by the record period command, PE.

The reported value is limited to +/- 2,147,483,647 decimal. The value is formatted with leading zeros and a minus character if negative.

Range: 0 <= ***n*** <= 1200

Registers: Position record point memory.

RAM based memory reserved of recording functions.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This two part routine reads back 1000
;     recorded optimal position points.

AL0+             ;Part 1, init. acc. <CR>

DO@0,+           ;Part 2, use acc. as index.
AA1,+            ;Add 1 to index value.
RP99+            ;Repeat 999 times. <CR>
```

Related Commands: **DQ, DR, DY, PE, PR**

DQ*n*

Display torque

This command reports the torque command level of the servo at time ***n***, relative to initiation of position record function, PR. The parameter ***n*** is an index to recorded data and is in units of multiple servo update periods as determined by the record period command, PE.

The reported value is limited to +/-1023 decimal. The value is formatted with leading zeros and a minus character if negative.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;    This two part routine reads back 1000
;    recorded torque command levels.

AL0+          ;Part 1, init. acc. <CR>

DQ@0,+        ;Part 2, use acc. as index.
AA1,+         ;Add 1 to index value.
RP99+         ;Repeat 999 times. <CR>
```

Range: 0 <= ***n*** <= 1200

Registers: Position record point memory.

RAM based memory reserved of recording functions.

Related Commands: **DO, DR, DY, PE, PR**

DR*n*

Display recorded

This command reports the actual, or "recorded", position of the servo at time ***n***, relative to initiation of position record function, PR. The parameter ***n*** is an index to recorded data and is in units of multiple servo update periods as determined by the record period command, PE.

The reported value is limited to +/-2,147,483,647 decimal. The value is formatted with leading zeros and a minus character if negative.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;    This two part routine reads back 1000
;    recorded servo position points.

AL0+          ;Part 1, init. acc. <CR>

DR@0,+        ;Part 2, use acc. as index.
AA1,+         ;Add 1 to index value.
RP99+         ;Repeat 999 times. <CR>
```

Range: 0 <= ***n*** <= 1200

Registers: Position record point memory.

RAM based memory reserved of recording functions.

Related Commands: **DO, DQ, DY, PE, PR**

DY*n*

Set record delay

This command sets a delay period of (***n*** * 0.5 mS) relative to initiation of the position record function, PR. When the PR command is executed, the DSC will wait (***n*** * 0.5 mS) and then start recording.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine initializes the PR function,
;       and then records 1000 data points.

PM,DH,+         ;Position mode, home pos.
DY100,+         ;Set 50 mS record delay.
PE2,+           ;Set 1 mS record period
MA40000,+       ;Start 10 revolution move.
PR1000+         ;Record 1 second of data.
```

Range: 0 <= ***n*** <= 32,767

Registers: none.

Related Commands: **DO, DQ, DR, PE, PR**

HE

Help

This command reports the list of commands supported by the installed DSC firmware. These commands are organized by group.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: none

Registers: none

Related Commands: **VE**

PE*n*

Set record period

This command sets a recording period of (***n*** * 0.5 mS) for the position record function, PR.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This routine initializes the PR function,
;     and then records 1000 data points.

PM,DH,+          ;Position mode, home pos.
DY100,+          ;Set 50 mS record delay.
PE2,+            ;Set 1 mS record period
MA40000,+        ;Start 10 revolution move.
PR1000+          ;Record 1 second of data.
```

Range: 1 <= ***n*** <= 32,767

Registers: none.

Related Commands: **DO, DQ, DR, DY, PR**

PR*n*

Position record enable

This command enables the position record function. When PR is executed, The DSC will delay recording as determined by the DY command, and then record a total of ***n*** data points. The time period between data points is determined by the PE command.

Each data point is stored in a 10 byte record consisting of: Optimal position, recorded position, and output torque command level. Data display commands, DO, DR and DQ are used to retrieve recorded data.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This routine switches to slave mode
;     and then records 1000 data points after
;     the servo's position has passed through 0.

DH-100,+         ;Initialize servo to -100 pos.
SM268435456,+    ;Slave mode, ratio = 1:1.
WP0,+            ;Wait for absolute position 0.
PR1000,+         ;Begin recording data.
WA2000+          ;Wait for pos. record to end.
```

Range: 1 <= ***n*** <= 1200

Registers: Servo Status, bit 15
Position record point memory

When the recording function is active, bit 15 of the Servo Status register is set. Recorded data is stored in a RAM based point memory table.

Related Commands: **DO, DQ, DR, DY, PE**

TA*n*

Tell analog

This command reports the present value of analog input number ***n***. Three analog inputs are supported by the DSC. Input 1 returns a value of +/-1023 counts, representing +/-10 Volts at the external input located on connector P4. Input 2 returns a value of 0 to 1023, representing the present magnitude of actual motor current. Scaling for input 2 is 32 counts/Amp. Input 3 returns a value of 0 to 1023 representing relative motor temperature. See DSC Hardware Reference, for more information.

All values are formatted with no leading zeros and a minus character if negative.

Range: ***n*** = 1: External user input.
= 2: Actual current magnitude.
= 3: Motor temperature.

Registers: none.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This is a command line example where TA
;     is entered at the DSC prompt.

>TA1          ;<CR>
126           ;A value +<CR> is returned,
>             ;and then a new prompt.
```

Related Commands: **GA**

TB

Tell breakpoint

This command reports the breakpoint position as determined by "Wait for ...", and "Interrupt on ...", position commands; WP, WR, IP, and IR.

The reported value is limited to +/-2,147,483,647 decimal. The value is formatted with leading zeros and a minus character if negative.

Range: none

Registers: Servo Table, offset 16.

This value is stored at Servo Table offset 16. This value can be copied to the accumulator by executing RL16<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This is a command line example where TB
;     is entered at the DSC prompt.

>TB           ;<CR>
0000004005    ;A value +<CR> is returned,
>             ;and then a new prompt.
```

Related Commands: **WP, WR, IP, IR**

TC*n*

Tell channel

This command reports the present state of channel number ***n***. 10 channels are supported by the DSC. See DSC System Overview and Hardware Reference for additional information.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range:

n =	1:	Output 0,	P4-9
=	2:	Output 1,	P4-10
=	3:	Output 2,	P4-11
=	4:	Output 3,	P4-12
=	5:	Input 0,	P3-7
=	6:	Input 1,	P3-9
=	7:	Input 2,	P3-10
=	8:	Input 3,	P3-11
=	9:	Input 4,	P3-12
=	10:	Fault Relay,	P3-1

Registers: none.

Example:

```
;     This two part routine reports the state of
;     all 10 channels.

AL1+          ;Part 1, init. acc. <CR>

TC@0,+        ;Part 2, use acc. as index.
AA1,+         ;Add 1 to index value.
RP9+          ;Repeat 9 times. <CR>
```

Related Commands: **CF, CN, WF, WN, DN, DF**

TD

Tell derivative gain

This command reports the present derivative gain setting as determined by the SD command.

The reported value is limited to 32,767 decimal and formatted with leading zeros.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: none.

Registers: Servo Table, offset 72.

This value is stored at Servo Table offset 72. This value can be copied to the accumulator by executing RW72<*CR*>.

Example:

```
;     This is a command line example where TD
;     is entered at the DSC prompt.

>TD           ;<CR>
0000004000    ;A value +<CR> is returned,
>             ;and then a new prompt.
```

Related Commands: **SD**

TE

Tell command error

This command reports the last command interpreter error code. Once this command is executed, the error code will be reset to 0, and the Servo Status register command error flag will be cleared.

Error codes are as follows:

-1 Parameter error.
-2 Command not valid.
-3 Negative repeat.
-4 Macro definition command not first.
-5 Macro number out of range.
-6 Macro does not exist.
-7 Command canceled.
-8 Axis number out of range.
-9 Motor on not allowed during command.

Range: none.

Registers: Servo Status, bit 22.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This is a command line example where TD
;     is entered at the DSC prompt.

>TE             ;<CR>
-1              ;A code +<CR> is returned,
>               ;and then a new prompt.
```

Related Commands: **CE**

TF

Tell following error

This command reports the present amount of servo following error. Following error is the encoder count difference between the servo's actual position and the servo's commanded position.

The reported value is limited to +/- 2,147,483,647 decimal. The value is formatted with leading zeros and a minus character if negative.

Range: none.

Registers: Servo Table, offset 82.

This value can be copied to the accumulator by executing RW82<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This is a command line example where TF
;     is entered at the DSC prompt.

>TF             ;<CR>
-0000000003     ;A value +<CR> is returned,
>               ;and then a new prompt.
```

Related Commands: **SE**

TG

Tell proportional gain

This command reports the present proportional gain setting as determined by the SG command.

The reported value is limited to 32,767 decimal and formatted with leading zeros.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This is a command line example where TG
;       is entered at the DSC prompt.

>TG             ;<CR>
0000001500      ;A code +<CR> is returned,
>               ;and then a new prompt.
```

Range: none.

Registers: Servo Table, offset 70.

This value is stored at Servo Table offset 70. This value can be copied to the accumulator by executing RW70<*CR*>.

Related Commands: **SG**

TI

Tell integral gain

This command reports the present integral gain setting as determined by the SI command.

The reported value is limited to 32,767 decimal and formatted with leading zeros.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This is a command line example where TI
;       is entered at the DSC prompt.

>TI             ;<CR>
0000000015      ;A value +<CR> is returned,
>               ;and then a new prompt.
```

Range: none.

Registers: Servo Table, offset 74.

This value is stored at Servo Table offset 74. This value can be copied to the accumulator by executing RW74<*CR*>.

Related Commands: **SI**

TL

Tell integration limit

This command reports the present integration limit setting as determined by the IL command.

The reported value is limited to 32,767 decimal and formatted with leading zeros.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This is a command line example where TL
;       is entered at the DSC prompt.

>TL             ;<CR>
0000000300      ;A code +<CR> is returned,
>               ;and then a new prompt.
```

Range: none.

Registers: Servo Table, offset 94.

This value is stored at Servo Table offset 94. This value can be copied to the accumulator by executing RW94<*CR*>.

Related Commands: **IL**

TO

Tell optimal

This command reports the present optimal, or "commanded", servo position. Optimal position is different from target position when the servo is in motion.

The reported value is limited to +\- 2,147,483,647 decimal. This value is formatted with leading zeros and a minus character if negative.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This is a command line example where TO
;       is entered at the DSC prompt.

>TO             ;<CR>
-0000032000     ;A value +<CR> is returned,
>               ;and then a new prompt.
```

Range: none.

Registers: Servo Table, offset 26.

This value can be copied to the accumulator by executing RL26<*CR*>.

Related Commands: **TT, TP**

TP

Tell position

This command reports the present, or "actual" servo position in units of quadrature encoder counts.

The reported value is limited to +\- 2,147,483,647 decimal. This value is formatted with leading zeros and a minus character if negative.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This is a command line example where TP
;       is entered at the DSC prompt.

>TP             ;<CR>
0000004400      ;A code +<CR> is returned,
>               ;and then a new prompt.
```

Range: none.

Registers: Servo Table, offset 4.

This value is stored at Servo Table offset 4. This value can be copied to the accumulator by executing RL4<*CR*>.

Related Commands: **DH, TO, TT**

TQ

Tell torque

This command reports the present commanded torque level.

The reported value is limited to +\- 1023 decimal. This value is formatted with leading zeros and a minus character if negative.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This is a command line example where TQ
;       is entered at the DSC prompt.

>TQ             ;<CR>
-0000000046     ;A value +<CR> is returned,
>               ;and then a new prompt.
```

Range: none.

Registers: Servo Table, offset 104.

This value can be copied to the accumulator by executing RW104<*CR*>.

Related Commands: **SQ**

TS

Tell servo status

This command reports the present state of the Servo Status register. The hex mode command, HM, can be used to view bits more directly.

Registers: Servo Table, offset 0.

The Servo Status register can be copied to the accumulator by executing RL0<*CR*>.

Example:

; This is a command line example.

```
>HM,TS,DM      ;Hex mode,TS,Dec. mode<CR>
0080422F       ;Hex value+<CR> is returned,
>              ;and then a new prompt.
```

Related Commands: **DM, HM**

Servo Status Bit Assignments:

Bit	Function	Bit	Function	Bit	Function	Bit	Function
0	Motor on	8	Encoder Phasing	16	Motor error	24	Limit mode abrupt
1	Trajectory complete	9	Position mode	17	Following error	25	Limit mode smooth
2	At target	10	Velocity mode	18	Amplifier error	26	Limit- tripped
3	Breakpoint reached	11	Torque mode	19	Motor over temp.	27	Limit- enabled
4	Position captured	12	Integration mode	20	Watchdog time out	28	Limit- active
5	Synchronization on	13	Contour mode	21	Reset input active	29	Limit+ tripped
6	Looking for index	14	Slave mode	22	Command error	30	Limit+ enabled
7	Servo Homed	15	Recording enabled	23	+5 Vdc brown out	31	Limit+ active

TT

Tell target

This command reports the target or "end" position for the last executed move command.

The reported value is limited to +\- 2,147,483,647 decimal. This value is formatted with leading zeros and a minus character if negative.

Range: none.

Registers: Servo Table, offset 20

This value can be copied to the accumulator by executing a RL20<*CR*>.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
(Note 1) Contour Mode - Yes

Note 1. In contour mode, TT reports the target point index value.

Example:

; This is a command line example where TT
; is entered at the DSC prompt.

```
>TT            ;<CR>
-0000044000    ;A value +<CR> is returned,
>              ;and then a new prompt.
```

Related Commands: **MA, MR, TO, TP**

TV

Tell command velocity

This command reports the magnitude of velocity presently commanded by the trajectory generator. The returned value is in units of: (quad. encoder counts / 0.5 mS * 65536).

The reported value is limited to 2,147,483,647 decimal and is formatted with leading zeros. See related commands for additional information.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - No
Contour Mode - No

Example:

```
;     This is a command line example where TV
;     is entered at the DSC prompt.

>TV             ;<CR>
0000104407      ;A code +<CR> is returned,
>               ;and then a new prompt.
```

Range: none.

Registers: Servo Table, offset 34.

This value is stored at Servo Table offset 34. This value can be copied to the accumulator by executing RL34<*CR*>.

Related Commands: **DS, SA, SV**

TX

Tell contour index

In contour mode, and when the servo is in motion, this command reports the present point index alone the contour.

The reported value is limited to 2047 decimal and is formatted with leading zeros.

Usage:

Position Mode - No
Velocity Mode - No
Gain Mode - No
Torque Mode - No
Slave Mode - No
Contour Mode - Yes

Example:

```
;     This is a command line example where TX
;     is entered at the DSC prompt.

>TX             ;<CR>
0000000046      ;A value +<CR> is returned,
>               ;and then a new prompt.
```

Range: none.

Registers: none.

Related Commands: **TT**

TZ

Tell servo index

This command reports the position where the index mark was found relative to the servo's position when the DSC was last reset or the DH command was executed.

The reported value is limited to +/- 2,147,483,647 decimal. This value is formatted with leading zeros and a minus character if negative.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This is a command line example where TZ
;     is entered at the DSC prompt.

>TZ                ;<CR>
0000003160         ;A code +<CR> is returned,
>                  ;and then a new prompt.
```

Range: none.

Registers: Servo Table, offset 8.

This value is stored at Servo Table offset 8. This value can be copied to the accumulator by executing RL8<*CR*>.

Related Commands: **DH, FI**

VE

Tell firmware version

This command reports the revision level of the firmware presently installed in the DSC

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This is a command line example where VE
;     is entered at the DSC prompt.

>VE                ;<CR>
DSC2000 Digital Servo Controller VER.AM1 REV 1.1
COPYRIGHT (c) Automotion Inc. 1996
ALL RIGHTS RESERVED
>                  ;and then a new prompt.
```

Range: none.

Registers: none.

Related Commands: **HE**

JP*n*

Jump to command absolute

This command causes the DSC command interpreter to jump to command number ***n*** within a macro command string. JP should not be used to jump into, or out of, a macro. Within a macro, commands are numbered consecutively as shown below:

```
MD8,+      ;Define macro 8, (RAM based)
JP@0,+     ;This is macro 8 command #0
TP,+       ;This is macro 8 command #1
TG,+       ;This is macro 8 command #2
TI,+       ;This is macro 8 command #3
TD+        ;This is macro 8 command #4
           ;<CR> to program this macro.
```

Range: 0 <= ***n*** <= 42

Registers: none.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;    This is a command line example to execute
;    only command numbers 3 and 4 of macro 8
;    (shown left).

>AL3          ;Load accumulator. <CR>
>MC8          ;Call macro number 8.<CR>
0000000005    ;Integrator value is reported.
0000006000    ;Derivative value is reported.
>             ;and then a new prompt.
```

Related Commands: **JR**

JR*n*

Jump to command relative

This command causes the DSC command interpreter to jump to relative command number ***n*** within a macro command string. JR should not be used to jump into, or out of, a macro. Within a macro, commands are numbered consecutively as shown below:

```
MD9,+      ;Define macro 9, (RAM based)
TP,+       ;Jump relative "-1" command here.
JR@0,+     ;Jump relative  "0" command here.
TG,+       ;Jump relative "+1" command here.
TI,+       ;Jump relative "+2" command here.
TD+        ;Jump relative "+3" command here.
           ;<CR> to program this macro.
```

Range: -42 <= ***n*** <= 42

Registers: none.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;    This is a command line example to execute
;    relative command number +3 of macro 9
;    (shown left), skipping over +1 and +2.

>AL3          ;Load accumulator. <CR>
>MC9          ;Call macro number 9.<CR>
0000010000    ;Servo's position is reported.
0000006000    ;Derivative value is reported.
>             ;and then a new prompt.
```

Related Commands: **JP**

MC*n*

Macro call

This command causes the DSC command interpreter to execute macro number ***n***. When execution of macro ***n*** is complete, the DSC command interpreter will return to the command immediately following the initiating macro call. Macros must first be defined before they can be called. See related commands.

Macro calls can be nested to create complex programs. A macro should not be programmed to call itself (i.e. recursion), unless commands have been included to limit the number of times it will do so.

Range: 0 <= ***n*** <= 255

Registers: none.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     Calling macro 4 below will initialize the
;     accumulator and then call macro 5, which
;     in turn, uses the accumulator as an index.

MD4,+       ;Define macro 4 as follows.
AL1,+       ;Load accumulator as index.
MC5+        ;Then call macro 5.

MD5,+       ;Define macro 5 as follows.
TC@0,+      ;Tell channel number = (index)
RP9         ;Loop macro 5, 9 times and end.
```

Related Commands: **MD, MJ, TM**

MD*n*

Macro define

This command defines an associated command string as macro number ***n***. Associated commands are those that follow the MD command, separated by commas, and that proceed a carriage return character as shown below:
MD5,<cmd>,<cmd>,<cmd>, ... <cmd>,<*CR*>

RAM based macros are available for scratch pad use. They can be redefined at any time by simply entering a new "MD***n***,..." command string.

When defining Flash based macros, the servo must be disabled using the MF command. Also, Flash based macros can only be programmed once, unless Flash memory is reset by executing: RM0<*CR*>. Also see "Stand Alone Operation, Macro 0" for additional information.

Range: 0 <= ***n*** <= 255

Registers: Macro storage

Macro numbers 1 to 10 are in RAM. Macro numbers 0, and 11 to 255 are stored in Flash.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This macro measures the servo's actual
;     velocity in units of encoder counts / 0.1 sec.

MD5,+       ;Define string as macro number 5.
RL4,+       ;Load accumulator w/ 1'st position.
AR10,+      ;Save 1'st position in register 10.
WA200,+     ;Wait for 0.1 seconds.
RL4,+       ;Load accumulator w/ 2'nd position.
AS@10,+     ;Subtract, (2'nd-1'st) = accumulator.
TR0+        ;Report value+<CR>, end macro 5.
```

Related Commands: **MC, RM, TM**

MJ*n*

Macro jump

This command may be used to "jump" to macro number ***n***. After execution of the MJ command, the DSC retains no record of where it jumped from. This means that commands following the MJ command will not be executed. If there is no macro defined by the number ***n***, an error will be reported.

The macro jump command can be used at any place in a command string or in a macro. It is also acceptable for a macro to jump to itself. This is an easy way to create polling or execution loops.

Range: 0 <= ***n*** <= 255

Registers: none.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     Calling macro 4 below will initialize the
;     accumulator, and then "jump" to the
;     macro pointed to by the accumulator.

MD4,+          ;Define macro 4 as follows.
AL5,+          ;Load accumulator as pointer.
MJ@0+          ;Then jump to macro <pointer>.

MD5,+          ;Define macro 5 as follows.
TP,WA200,+     ;Tell position, wait 100 mS.
RP9+           ;Loop macro 5, 9 times and end.
```

Related Commands: **MC, RP**

RM*n*

Reset memory

Used to initialize the memory space reserved for macro commands and contour point memory. Execution of RM0 is the only way Flash based macro commands and contour points may be removed after they are defined. The servo must be disabled (i.e. MF) before resetting Flash memory.

Once executed, RM0 removes all macro commands currently defined, so those macro commands still desired must be reprogrammed. It is always a good idea to use the RM command before programming or downloading application software.

Range: ***n*** = 0: Reset RAM and Flash memory
= 1: Reset RAM memory only
= 2: Reset Flash Contour Points only

Registers: Macro and Contour Point memory

Macro numbers 1 to 10 are in RAM. Macro numbers 0, 11 to 255, and Contour Points are stored in Flash.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Related Commands: **MD, TM**

TM*n*

Tell macro

This command may be used to list macro number ***n***. If ***n*** = -1, then all presently defined macros will be listed.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: -1 <= ***n*** <= 255

Registers: none.

Related Commands: **HE, MD, RM, VE**

BC*n*

Breakpoint channel

This command selects an output channel number, ***n***, that will be turned off upon execution of a IP, IR, WP or WR command, and turned on when the corresponding breakpoint position is reached. Note that channel output logic can be inverted by using the CH and CL commands.

See "DSC Hardware Reference - Mixed I/O, Output 0 to 3" for additional information.

Range: ***n*** = 0: No output selected
= 1: Output 0, P4-9
= 2: Output 1, P4-10
= 3: Output 2, P4-11
= 4: Output 3, P4-12

Registers: Servo Table, offset 112

This value is stored at Servo Table offset 112. This value can be copied to the accumulator by executing RB112<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;   This routine creates output sequencing based
;   on relative servo position.

BC2,+          ;Set out1 as breakpoint channel.
CN1,+          ;Turn output 0 on before move.
MR40000,+      ;Start relative 40000 count move
WR-20000,+     ;Out1 off, wait pos=(target-20000)
BC1,+          ;Out1 now on, set out0 as brk. pnt.
WR-10000,+     ;Out0 off, wait pos=(target-10000)
WS2000,+       ;Out0 now on, wait move end+1s.
RP9+           ;Repeat command string 9 times.
```

Related Commands: CH, CL, IP, IR, WP, WR

CF*n*

Channel off

This command turns output channel ***n*** off. "Off" is relative to "channel high..." or "channel low true logic" commands, CH and CL. If using channel high true logic, CF***n*** will set an open collector output channel to a high impedance state. Conversely, if using channel low true logic, CF***n*** will cause an open collector output channel to sink current.

See "DSC Hardware Reference" for additional information regarding

Range: ***n*** = 1: Output 0, P4-9
= 2: Output 1, P4-10
= 3: Output 2, P4-11
= 4: Output 3, P4-12
= 10: Fault Relay, P3-1

Registers: none.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;   This routine creates an output representing
;   motion active.

CH2,CF2,+      ;High true logic, set output 4 off.
MR40000,+      ;Start relative 40000 count move
WS50,+         ;Wait for move end+25 msec.
CN2+           ;Now turn output 4 on.
```

Related Commands: BC, CH, CL, CN

CH*n*

Channel high true logic

This command specifies output channel number ***n*** as having "high true" logic. Execution of CH***n*** will not cause output channel number ***n*** to change state. Rather, execution of CH***n*** means that subsequent channel commands, CF, CN and TC, will consider "on" as equal to 1, and "off" as equal to "0". Also, an "on" open collector output will sink current, and an "off" open collector output will be in a high impedance state.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: ***n*** = 1: Output 0, P4-9
= 2: Output 1, P4-10
= 3: Output 2, P4-11
= 4: Output 3, P4-12

Registers: none.

Related Commands: **CF, CN, TC**

CL*n*

Channel low true logic

This command specifies output channel number ***n*** as having "low true" logic. Execution of CL***n*** will not cause output channel number ***n*** to change state. Rather, execution of CL***n*** means that subsequent channel commands, CF, CN and TC, will consider "on" as equal to 0, and "off" as equal to "1". Also, an "on" open collector output will be in a high impedance state, and an "off" open collector output will sink current.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: ***n*** = 1: Output 0, P4-9
= 2: Output 1, P4-10
= 3: Output 2, P4-11
= 4: Output 3, P4-12

Registers: none.

Related Commands: **CF, CN, TC**

CN*n*

Channel on

This command turns output channel ***n*** on. "On" is relative to "channel high..." or "channel low true logic" commands, CH and CL. If using channel high true logic, CN***n*** will cause an open collector output channel to sink current. Conversely, if using channel low true logic, CN***n*** will set an open collector output channel to a high impedance state.

See "DSC Hardware Reference" for additional information regarding

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine creates an output representing
;       motion active.

CH2,CF2,+       ;High true logic, set output 4 off.
MR40000,+       ;Start relative 40000 count move
WS50,+          ;Wait for move end+25 msec.
CN2+            ;Now turn output 4 on.
```

Range:

n =	1:	Output 0,	P4-9
=	2:	Output 1,	P4-10
=	3:	Output 2,	P4-11
=	4:	Output 3,	P4-12
=	10:	Fault Relay,	P3-1

Registers: none.

Related Commands: **BC, CH, CL, CN**

DF*n*

Do if channel off

This command is used for conditional execution of a command string. If the specified input/output channel is "off", commands following DF***n*** in a command string, or in a macro, will be executed, else the rest of the command string, or macro, will be skipped.

For DSC outputs 0 to 3 (channels 1 to 4), "on" and "off" are relative to channel commands, CH and CL. See related commands for additional information.

Range: ***n*** = 1: Output 0, P4-9
= 2: Output 1, P4-10
= 3: Output 2, P4-11
= 4: Output 3, P4-12
= 5: Input 0, P3-7
= 6: Input 1, P3-9
= 7: Input 2, P3-10
= 8: Input 3, P3-11
= 9: Input 4, P3-12
= 10: Fault Relay, P3-1

Registers: none.

Usage: Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This routine reports servo position or servo
;     following error as determined by input 0.

MD4,DF5,TP+     ;In 0="off", tell pos.<CR>

MD5,DN5,TF+     ;In 0="on",tell follow<CR>

MC4,MC5,+       ;Tell as function of input 0.
WA400,RP+       ;Wait 200 mSec, loop <CR>
```

Related Commands: **CF, CH, CL, CN, TC**

DN*n*

Do if channel on

This command is used for conditional execution of a command string. If the specified input/output channel is "on", commands following DN***n*** in a command string, or in a macro, will be executed, else the rest of the command string, or macro, will be skipped.

For DSC outputs 0 to 3 (channels 1 to 4), "on" and "off" are relative to channel commands, CH and CL. See related commands for additional information.

Range: ***n*** = 1: Output 0, P4-9
= 2: Output 1, P4-10
= 3: Output 2, P4-11
= 4: Output 3, P4-12
= 5: Input 0, P3-7
= 6: Input 1, P3-9
= 7: Input 2, P3-10
= 8: Input 3, P3-11
= 9: Input 4, P3-12
= 10: Fault Relay, P3-1

Registers: none.

Usage: Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This routine reports servo position or servo
;     following error as determined by input 0.

MD4,DF5,TP+     ;In 0="off", tell pos.<CR>

MD5,DN5,TF+     ;In 0="on",tell follow<CR>

MC4,MC5,+       ;Tell as function of input 0.
WA400,RP+       ;Wait 200 mSec, loop <CR>
```

Related Commands: **CF, CH, CL, CN, TC**

DT*n*

Delay at target

This command sets bit 2 in the servo status register after the servo's actual position is within the target range, as defined by the DB command, for a period of: ***n****0.5 mSec. Bit 2 of the servo status register can be evaluated as required within a user's program. In this way, the user can determine that a move has been completed to practical level of precision for a specified time.

See related commands for further information.

Range: 0 <= ***n*** < = 32,767

Registers: Servo Table, offset 108
Servo status, bit 2

When the DT command is executed, the value ***n*** is stored in the RAM based Servo Table at offset 108, decimal. The value ***n*** can be accessed by executing RW108,TR0<*CR*>. The default value is zero. Bit 2 of the Servo status register can be viewed by executing HM,TS,DM<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - No
Contour Mode - No

Example:

```
;    This routine makes a 12000 count move,
;    and reports actual position after the servo
;    is within target range for 25 mSec.

DB10,DT50+      ;Set deadband and delay.
MR12000,+       ;Start move
WT0,+           ;Wait for servo at target.
TP+             ;Report position.
```

Related Commands: **DB, WT**

IB*n*

IF accumulator is below

This command is used for conditional execution within a command string. IB***n*** is interpreted as follows:

IF accumulator < ***n*** Then Do next command,
Else skip 2 commands.

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: User Register 0, accumulator.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;    This routine waits for a positive reading
;    from the analog input on connector P4.

GA1,+       ;Copy analog input to accumulator.
IB0,+       ;If input is negative do next 2
UD118,+     ;Display an "H" for status.
RP,+        ;Loop to first string command.
UD0+        ;Else, clear status and continue
```

Related Commands: **IE, IG**

IC*n*

If accumulator bit is clear

This command is used for conditional execution within a command string. IC***n*** is interpreted as follows:

IF accum. bit ***n*** = 0 Then Do next command,
Else skip 2 commands.

Where: Bit 0 is lest significant.

Range: 0 < = ***n*** < =31

Registers: User Register 0, accumulator.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine positions the servo to the limit+
;       input active point.

UD128,+     ;Light Status display decimal point
RL0,+       ;Copy Servo Status to accumulator.
IC31,+      ;If limit+ not active, do next 2
MR1,+       ;Move a little closer to limit..
RP,+        ;Loop to first string command.
UD0+        ;Else, clear status and continue
```

Related Commands: **IS**

IE*n*

IF accumulator is equal

This command is used for conditional execution within a command string. IE***n*** is interpreted as follows:

IF accumulator = ***n*** Then Do next command,
Else skip 2 commands.

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: User Register 0, accumulator.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine checks for all fault bits in the
;       Servo Status register.

RL0,+           ;Servo Status to accumulator.
AN16711680,+    ;Accumulator AND FF0000 hex.
IE0,+           ;If bits 16 to 23 not set do next 2.
UD0,+           ;Status OK, clear status display
MJ20,+          ;Jump to program, macro 20.
TR0,+           ;Else, report fault status.
RP+             ;loop until <ESC> received.
```

Related Commands: **IU**

IF*n*

If channel is off

This command is used for conditional execution within a command string:

IF channel ***n*** = "off" Then Do next command,
Else skip 2 commands.

For DSC outputs 0 to 3 (channels 1 to 4), "on" and "off" are relative to channel commands, CH and CL. See related commands for additional information.

Range:

n =	1:	Output 0,	P4-9
=	2:	Output 1,	P4-10
=	3:	Output 2,	P4-11
=	4:	Output 3,	P4-12
=	5:	Input 0,	P3-7
=	6:	Input 1,	P3-9
=	7:	Input 2,	P3-10
=	8:	Input 3,	P3-11
=	9:	Input 4,	P3-12
=	10:	Fault Relay,	P3-1

Registers: none.

Usage: Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This routine uses the Status display to show
;     the state of input number 4.

IN9,+          ;If in 4 ="off", Do next 2
UD63,+         ;Update Status display to "0".
RP,+           ;Loop to start of string.
UD6,+          ;Else set Status display to "1".
RP+            ;Loop to start of string.
```

Related Commands: **CH, CL, DF, DN, IN**

IG*n*

IF accumulator is greater

This command is used for conditional execution within a command string. IG***n*** is interpreted as follows:

IF accumulator > ***n*** Then Do next command,
Else skip 2 commands.

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: User Register 0, accumulator.

Usage: Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This routine checks for all fault bits in the
;     Servo Status register.

RL0,+          ;Servo Status to accumulator.
AN16711680,+   ;Accumulator AND FF0000 hex.
IG0,+          ;If bits 16 to 23 are set do next 2.
TR0,+          ;Report fault code from status.
RP,+           ;loop until <ESC> received.
UD0+           ;Else, clear status display
```

Related Commands: **IB, IE**

IN*n*

If channel is on

This command is used for conditional execution within a command string:

IF channel ***n*** = "on" Then Do next command,
Else skip 2 commands.

For DSC outputs 0 to 3 (channels 1 to 4), "on" and "off" are relative to channel commands, CH and CL. See related commands for additional information.

Range:

n =	1:	Output 0,	P4-9
=	2:	Output 1,	P4-10
=	3:	Output 2,	P4-11
=	4:	Output 3,	P4-12
=	5:	Input 0,	P3-7
=	6:	Input 1,	P3-9
=	7:	Input 2,	P3-10
=	8:	Input 3,	P3-11
=	9:	Input 4,	P3-12
=	10:	Fault Relay,	P3-1

Registers: none.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This routine uses the status display to show
;     the state of input number 4.

IN9,+          ;If in 4 ="on", Do next 2
UD6,+          ;Update Status display to "1".
RP,+           ;Loop to start of string.
UD63,+         ;Else set Status display to "0".
RP+            ;Loop to start of string.
```

Related Commands: **CH, CL, DF, DN, IF**

IP*n*

Interrupt on absolute position

This command is used to indicate that the servo has reached absolute position ***n*** . Absolute position is relative to home position.

Upon execution of IP***n***, the DSC will reset bit 3 in the Servo Status register and turn "off" the specified breakpoint output channel. The DSC will then continue to interpret any commands that follow. When the servo has reached the specified position, the DSC will set bit 3 in the servo status register and turn "on" the specified breakpoint output channel.

The IP command can be executed before or after motion is commanded.

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: Servo Table, offset 16
Servo Status, bit 3

The value ***n*** is stored at Servo Table offset 16. This value can be copied to the accumulator by executing RL16<*CR*>. Servo Status, bit 3 is "breakpoint reached."

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This routine creates output sequencing based
;     on absolute servo position.

BC2,DH0,+      ;Set break channel, define home.
MA40000,+      ;Start absolute 40000 count move.
IP16000,+      ;Bit 3=0, Out1 off, int. pos=16000.
HM,+           ;Switch to hex mode for reporting.
TS,+           ;Report status while moving.
WS0,+          ;Wait for move end+0 seconds.
TS,DM+         ;Report status, set to decimal mode.
```

Related Commands: BC, CH, CL, IR, WP, WR

IR*n*

Interrupt on relative position

This command is used to indicate that the servo has reached position ***n*** relative to a move target position.

Upon execution of IR***n***, the DSC will reset bit 3 in the Servo Status register and turn "off" the specified breakpoint output channel. The DSC will then continue to interpret any commands that follow. When the servo has reached the specified position, the DSC will set bit 3 in the Servo Status register and turn "on" the specified breakpoint output channel.

The IR command can be executed before or after motion is commanded.

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: Servo Table, offset 16
Servo Status, bit 3

The value (move target + ***n***) is stored at Servo Table offset 16. This value can be copied to the accumulator by executing RL16<*CR*>. Servo Status, bit 3 is "breakpoint reached."

Usage: (Note 1)
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - No
Contour Mode - No

Note 1. Only when stopping via the ST command.

Example:

```
;       This routine creates output sequencing based
;       on relative servo position.

BC2,+           ;Set breakpoint channel as Out 1.
MR40000,+       ;Start relative 40000 count move.
IR-999,+        ;Bit 3=0, Out1 off, pos=target-999.
HM,+            ;Switch to hex mode for reporting.
TS,+            ;Report status while moving.
WS0,+           ;Wait for move end+0 seconds.
TS,DM+          ;Report status, set to decimal mode.
```

Related Commands: BC, CH, CL, IP, WP, WR

IS*n*

If accumulator bit is set

This command is used for conditional execution within a command string. IS***n*** is interpreted as follows:

IF accum. bit ***n*** = 1 Then Do next command,
Else skip 2 commands.

Where: Bit 0 is lest significant.

Range: 0 < = ***n*** < = 31

Registers: User Register 0, accumulator.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       After a system reset has occurred, this
;       routine checks to see if it was induced by
;       a Watchdog time out.

RL0,+           ;Copy Servo Status to accumulator.
IS20,+          ;If watchdog bit set, do next 2
TS,+            ;Report status register on fault
RP,+            ;Loop to first string command.
NO+             ;Else, continue.
```

Related Commands: **IC**

IU*n*

IF accumulator is unequal

This command is used for conditional execution within a command string. IU***n*** is interpreted as follows:

IF accum. < or > ***n*** Then Do next command,
Else skip 2 commands.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine checks for all fault bits in the
;       Servo Status register.

RL0,+           ;Servo Status to accumulator.
AN16711680,+    ;Accumulator AND FF0000 hex.
IU0,+           ;If bits 16 to 23 set do next 2.
TR0,+           ;Report fault status.
RP,+            ;loop until <ESC> received.
UD0+            ;Clear status display & continue.
```

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: User Register 0, accumulator.

Related Commands: **IE**

RP*n*

Repeat command string

This command tells the DSC to ececute the proceeding command string, ***n*** more times. A value of ***n*** = 0 will tell the DSC to repeat indefinitely.

Nesting of RP commands within a command string is not allowed. Nested loops can be created by nesting macro calls, and then repeating within macros.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine reports servo position 100 times

TP,+            ;Report servo postion.
WA500,+         ;Wait for 250 mSec.
RP99,+          ;Repeat 99 times.
```

Range: 0 <= ***n*** <= 65,535

Registers: User Register 0, accumulator.

Related Commands: **MC**

WA*n*

Wait for time

This command suspends operation of the DSC command interpreter for a period equal to: ***n****0.5 mSec.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine reports servo position 100 times

TP,+            ;Report servo postion.
WA500,+         ;Wait for 250 mSec.
RP99+           ;Repeat 99 times.
```

Range: 1 <= ***n*** <= 65,535

Registers: none.

Related Commands: **WS, WT**

WF*n*

Wait for channel off

This command suspends operation of the DSC command interpreter until input channel number ***n*** is off.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine waits for a rising edge
;       transition at input number 4.

UD6,+           ;Set Status display to "1".
WF9,+           ;Wait until input 4 is "off".
UD63,+          ;Set Status display to "0".
WA40,+          ;Wait for 20 mSec, debounce.
WN9,+           ;Wait until input 4 is "on".
UD6+            ;Set Status display to "1".
```

Range:

n =	5:	Input 0,	P3-7
=	6:	Input 1,	P3-9
=	7:	Input 2,	P3-10
=	8:	Input 3,	P3-11
=	9:	Input 4,	P3-12

Registers: none.

Related Commands: **WN**

WN*n*

Wait for channel on

This command suspends operation of the DSC command interpreter until input channel number ***n*** is on.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine waits for a rising edge
;       transition at input number 4.

UD6,+           ;Set Status display to "1".
WF9,+           ;Wait until input 4 is "off".
UD63,+          ;Set Status display to "0".
WA40,+          ;Wait for 20 mSec, debounce.
WN9,+           ;Wait until input 4 is "on".
UD6+            ;Set Status display to "1".
```

Range:

n =	5:	Input 0,	P3-7
=	6:	Input 1,	P3-9
=	7:	Input 2,	P3-10
=	8:	Input 3,	P3-11
=	9:	Input 4,	P3-12

Registers: none.

Related Commands: **WF**

WP*n*

Wait for absolute position

This command suspends operation of the DSC command interpreter until the servo has reached absolute position ***n***. Absolute position is relative to home position.

Upon execution of WP***n***, the DSC will reset bit 3 in the Servo Status register and turn "off" the specified breakpoint output channel. The DSC will then suspend command interpretation. When the servo has reached the specified position, the DSC will set bit 3 in the servo status register, turn "on" the specified breakpoint output channel, and resume command interpretation.

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: Servo Table, offset 16
Servo Status, bit 3

The value ***n*** is stored at Servo Table offset 16. This value can be copied to the accumulator by executing RL16<*CR*>. Servo Status, bit 3 is "breakpoint reached."

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine creates output sequencing based
;       on absolute servo position.

BC2,DH0,+       ;Set break channel, define home.
MA40000,+       ;Start absolute 40000 count move.
WP16000,+       ;Bit 3=0, Out1 off, wait pos=16000.
HM,+            ;Switch to hex mode for reporting.
TS,+            ;Report status while moving.
WS0,+           ;Wait for move end+0 seconds.
TS,DM+          ;Report status, set to decimal mode.
```

Related Commands: CH, CL, BC, IP, IR, WR

WR*n*

Wait for relative position

This command suspends operation of the DSC command interpreter until the servo has reached position ***n*** relative to a move target position.

Upon execution of WR***n***, the DSC will reset bit 3 in the Servo Status register, turn "off" the specified breakpoint output channel and then suspend command interpretation. When the servo has reached the specified position, the DSC will set bit 3 in the servo status register, turn "on" the specified breakpoint output channel and resume command interpretation.

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: Servo Table, offset 16
Servo Status, bit 3

The value (move target **+ *n***) is stored at Servo Table offset 16. This value can be copied to the accumulator by executing RL16<*CR*>. Servo Status bit 3 is "breakpoint reached."

Usage: (Note 1)
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - No
Contour Mode - No

Note 1. Only when stopping via the ST command.

Example:

```
;     This routine creates output sequencing based
;     on absolute servo position.

BC2,DH,+     ;Breakpoint chan=Out 1,def. home.
MA-4000,+    ;Start absolute 4000 count move.
WR1500,+     ;Bit 3=0, Out1 off, wait pos=-2500.
HM,+         ;Switch to hex mode for reporting.
TS,+         ;Report status while moving.
WS0,+        ;Wait for move end+0 seconds.
TS,DM+       ;Report status, set to decimal mode.
```

Related Commands: CH, CL, BC, IP, IR, WP

WS*n*

Wait for stop

This command suspends operation of the command interpreter until the trajectory generator has completed a move, and an additional period of ***n****0.5 milliseconds has elapsed.

Range: 0 <= ***n*** <= 65,535

Registers: Servo Status, bit 1

Bit 1 of the Servo Status register indicates "Trajectory complete".

Usage: (Note 1)
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - No
Torque Mode - No
Slave Mode - No
Contour Mode - No

Note 1. Only when stopping via the ST command.

Example:

```
;     This routine reports actual servo position
;     shortly after a move is complete.

MR4000,+     ;Start relative 4000 count move.
WS50,+       ;Wait for move end + 25 mSec.
TP+          ;Report actual servo position.
```

Related Commands: WT

WT*n*

Wait for target

This command suspends operation of the command interpreter until the servo is "At target", and an additional period of ***n****0.5 milliseconds has elapsed. See related commands for specifying target deadband and delay.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - No
Slave Mode - No
Contour Mode - No

Example:

```
;       This routine makes a 12000 count move,
;       and reports actual position after the servo
;       is within target range for 25 mSec.

DB10,DT50+      ;Set deadband and delay.
MR12000,+       ;Start move
WT0,+           ;Wait for at target + 0 mSec.
TP+             ;Report position.
```

Range: 0 <= ***n*** < = 65,535

Registers: Servo Status, bit 2

Bit 2 of the Servo Status register indicates "At target".

Related Commands: **DB, DT**

LP*n*

Learn point

This command programs the servo's present position as contour mode point number ***n***. Contour points are stored in Flash memory and are write once. The RM2 command clears contour mode point memory.

See "DSC Servo Control, Contour Mode" for additional information.

Usage:
Position Mode - Yes
Velocity Mode - No
Gain Mode - Yes
Torque Mode - No
Slave Mode - No
Contour Mode - No

Range: 0 <= ***n*** <= 2047

Registers: Servo Table, offset 4

The servo's present position is stored at Servo Table offset 4. This value can be copied to the accumulator by executing RL4<*CR*>.

Related Commands: **CM, LT, MA, MR, RM**

LT*n*

Learn target

This command programs the present move target position as contour mode point number ***n***. Contour points are stored in Flash memory and are write once. The RM2 command clears contour mode point memory.

See "DSC Servo Control, Contour Mode" for additional information.

Usage:
Position Mode - Yes
Velocity Mode - No
Gain Mode - Yes
Torque Mode - No
Slave Mode - No
Contour Mode - No

Range: 0 <= ***n*** <= 2047

Registers: Servo Table, offset 20

The present move target position is stored at Servo Table offset 20. This value can be copied to the accumulator by executing RL20<*CR*>.

Examples:

```
;       This routine programs points 0 to 4 in
;       contour mode point memory.

PM,MF,RM2,+       ;Pos Mode, reset points.
DH,LT0,+          ;Def. home, learn point 0.
MR10000,LT1,+     ;Set and learn point 1.
MR-5000,LT2,+     ;Set and learn point 2.
MR,5000,LT3,+     ;Set and learn point 3.
MR-10000,LT4+     ;Set and learn point 4.
```

Related Commands: **CM, LP, MA, MR, RM**

AA*n*

Accumulator add

This command adds the value ***n*** to the accumulator. If the resulting value is larger than 32 bits, the most significant bit will be lost. AA***n*** is interpreted as follows:

accumulator = accumulator + ***n***

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: User Register 0, accumulator.

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This two part routine reads back 1000
;       recorded servo position points.

AL0+            ;Part 1, init. acc. <CR>

DR@0,+          ;Part 2, use acc. as index.
AA1,+           ;Add 1 to index value.
RP99+           ;Repeat 999 times. <CR>
```

Related Commands: **AD, AM, AS**

AC

Accumulator compliment

This command logically complements the accumulator in bit wise form. AC is interpreted as follows:

accumulator = **!**accumulator

Range: none.

Registers: User Register 0, accumulator.

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This is a command line example.

>HM,AL1,TR0     ;Enter string+<CR> at prompt.
00000001        ;Acc in hex format is returned.
>AC,TR0,DM      ;Enter string+<CR> at prompt.
FFFFFFFE        ;!Acc, hex format, is returned.
```

Related Commands: **AE, AN, AO**

AD*n*

Accumulator divide

This command performs 64 bit by 32 bit signed division with a 32 bit result and a 32 bit remainder. Before execution of AD, the lowest order 32 bits of the numerator must be in placed in the accumulator, and the highest order 32 bits must be placed in user register 1. The result is returned in the accumulator and the remainder is returned in user register 1. AD***n*** is interpreted as follows:

accumulator = {reg.1 & accumulator} / ***n***
user reg. 1 = {remainder}

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: User Register 0, accumulator
User Register 1

The contents of the accumulator may be viewed by executing TR0<*CR*>. The contents of register 1 may be viewed by executing TR1<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine divides 7 by 2.

AL0,AR1,+       ;Initialized high order bits to 0.
AL7,+           ;Initialize low order bits to "7".
AD2,+           ;Divide by "2".
TR0,+           ;Tell result.
TR1+            ;Tell remainder.
```

Related Commands: **AA, AM, AS**

AE*n*

Accumulator exclusive OR

This command performs a logical exclusive OR with the accumulator and the value ***n***. AC is interpreted as follows:

accumulator = accumulator XOR ***n***

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This is a command line example.

>HM,AL5,TR0     ;Enter string+<CR> at prompt.
00000005        ;Acc in hex format is returned.
>AE1,TR0,DM     ;Enter string+<CR> at prompt.
00000004        ;5 XOR 1 = 4.
```

Related Commands: **AC, AN, AO**

AL*n*

Accumulator load

This command loads the accumulator with the value ***n***. AL***n*** is interpreted as follows:

accumulator = ***n***

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This two part routine reads back 1000
;     recorded servo position points.

AL0+          ;Part 1, init. acc. <CR>

DR@0,+        ;Part 2, use acc. as index.
AA1,+         ;Add 1 to index value.
RP99+         ;Repeat 999 times. <CR>
```

Related Commands: **AR, RA**

AM*n*

Accumulator multiply

This command performs 32 bit by 32 bit signed multiply with a 64 bit result. The lowest order 32 bits of the result is placed in the accumulator, and the highest order 32 bits of the result is placed in user register 1. AM***n*** is interpreted as follows:

{reg.1 & accumulator} = accumulator * ***n***

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: User Register 0, accumulator
User Register 1

The contents of the accumulator may be viewed by executing TR0<*CR*>. The contents of register 1 may be viewed by executing TR1<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This routine multiplies 7 by 2.

AL7,+         ;Initialize accumulator to "7".
AM2,+         ;Multiply accumulator by "2".
TR1,+         ;Tell high order bits of result.
TR0+          ;Tell low order bits of result.
```

Related Commands: **AA, AD, AS**

AN*n*

Accumulator AND

This command performs a logical AND with the accumulator and the value ***n***. AN is interpreted as follows:

accumulator = accumulator AND ***n***

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

; This is a command line example.

```
>HM,AL5,TR0     ;Enter string+<CR> at prompt.
00000005        ;Acc in hex format is returned.
>AN1,TR0,DM     ;Enter string+<CR> at prompt.
00000001        ;5 AND 1 = 1.
```

Related Commands: **AC, AE, AO**

AO*n*

Accumulator OR

This command performs a logical OR with the accumulator and the value ***n***. AO is interpreted as follows:

accumulator = accumulator OR ***n***

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

; This is a command line example.

```
>HM,AL5,TR0     ;Enter string+<CR> at prompt.
00000005        ;Acc in hex format is returned.
>AE1,TR0,DM     ;Enter string+<CR> at prompt.
00000005        ;5 OR 1 = 5.
```

Related Commands: **AC, AN, AE**

AR*n*

Copy accumulator to register

This command copies the accumulator to user register ***n***. AR***n*** is interpreted as follows:

register ***n*** = accumulator

Range: 1 <= ***n*** <= 255

Registers: User Register 0, accumulator
User Register ***n***

The contents of the accumulator may be viewed by executing TR0<*CR*>. The contents of register ***n*** may be viewed by executing TR***n***<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine divides 7 by 2.

AL0,AR1,+       ;Initialized high order bits to 0.
AL7,+           ;Initialize low order bits to "7".
AD2,+           ;Divide by "2".
TR0,+           ;Tell result.
TR1+            ;Tell remainder.
```

Related Commands: **AL, RA**

AS*n*

Accumulator subtract

This command subtracts the value ***n*** from the accumulator. If the resulting value is larger than 32 bits, the most significant bit will be lost. AS***n*** is interpreted as follows:

accumulator = accumulator - ***n***

Range: -2,147,483,647 < ***n*** < 2,147,483,647

Registers: User Register 0, accumulator.

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine subtracts 100 from 0.

AL0,+           ;Part 1, init. acc. <CR>
AS100,+         ;Part 2, use acc. as index.
TR0+            ;Add 1 to index value.
```

Related Commands: **AA, AD, AM**

GA*n*

Copy analog input to accumulator

This command copies analog input ***n*** to the accumulator. Three analog inputs are supported by the DSC. Input 1 returns a value of +/-1023 counts, representing +/-10 Volts at the external input located on connector P4. Input 2 returns a value of 0 to 1023, representing the present magnitude of actual motor current. Scaling for input 2 is 32 counts/Amp. Input 3 returns a value of 0 to 1023 representing relative motor temperature. See DSC Hardware Reference for more information.

GA***n*** is interpreted as follows:

accumulator = analog input ***n***

Range: ***n*** = 1: External user input.
= 2: Actual current magnitude.
= 3: Motor temperature.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine checks motor temperature.

UD128,+     ;Light status display for test.
GA3,+       ;Get motor temp.
IG750,+     ;IF value>750 do next 2 commands.
MF,+        ;Turn motor off.
RP,+        ;Loop until temp is OK.
UD0+        ;Temp is OK, display off, continue.
```

Related Commands: **TA**

GB*n*

Copy byte to accumulator

This command copies a byte from physical memory address ***n*** to the accumulator. GB***n*** is interpreted as follows:

accumulator = Byte @ address ***n***

Range: 41216(A100hex)<= ***n*** <=53243(CFFBhex)

Note: Range given is RAM based position recording point memory. If not using the PR function, then this memory is available for general use.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Related Commands: **GW, GL, PB, PW, PL**

GL*n*

Copy long word to accumulator

This command copies a byte from physical memory address ***n*** to the accumulator. GL***n*** is interpreted as follows:

accumulator = long word @ address ***n***

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: 41216(A100hex)<= ***n*** <=53243(CFFBhex)

Note: Range given is RAM based position recording point memory. If not using the PR function, then this memory is available for general use.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Related Commands: **GB, GW, PB, PW, PL**

GN

Copy node switch to accumulator

This command copies the DSC node switch value to the accumulator. Switch 1 is the least significant bit and switch 5 is the most significant bit. GN is interpreted as follows:

accumulator = node switch value.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;     This routine displays the binary value of the
;     node switch on the DSC status display.

GN,+          ;Copy node switch to acc.
AN15,+        ;Mask of switch 5.
UD@0,+        ;Write to status display.
RP+           ;Loop string, <ESC> to end.
```

Range: none.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Related Commands: **GS, UD**

GS

Copy select switch to accumulator

This command copies the DSC select switch value to the accumulator. This is a 4 bit binary value corresponding to the switch setting 0 to 9. GS is interpreted as follows:

accumulator = select switch value.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine displays the binary value of the
;       select switch on the DSC status display.

GS,+            ;Copy node switch to acc.
UD@0,+          ;Write to status display.
RP+             ;Loop string, <ESC> to end.
```

Range: none.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Related Commands: **GN, UD**

GW*n*

Copy word to accumulator

This command copies a word from physical memory address ***n*** to the accumulator. GW***n*** is interpreted as follows:

accumulator = word @ address ***n***

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: 41216(A100hex)<= ***n*** <=53243(CFFBhex)

Note: Range given is RAM based position recording point memory. If not using the PR function, then this memory is available for general use.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Related Commands: **GB, GL, PB, PW, PL**

PB*n*

Copy accumulator to byte

This command copies the accumulator to a byte located at physical memory address ***n***. PB***n*** is interpreted as follows:

byte @ address ***n*** = accumulator

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: 41216(A100hex)<= ***n*** <=53243(CFFBhex)

Note: Range given is RAM based position recording point memory. If not using the PR function, then this memory is available for general use.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Related Commands: **GB, GL, GW, PW, PL**

PL*n*

Copy accumulator to long word

This command copies the accumulator to a long word located at physical memory address ***n***. PL***n*** is interpreted as follows:

long word @ address ***n*** = accumulator

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: 41216(A100hex)<= ***n*** <=53243(CFFBhex)

Note: Range given is RAM based position recording point memory. If not using the PR function, then this memory is available for general use.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Related Commands: **GB, GL, GW, PB, PW**

PW*n*

Copy accumulator to word

This command copies the accumulator to a word located at physical memory address ***n***. PW***n*** is interpreted as follows:

word @ address ***n*** = accumulator

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: 41216(A100hex)<= ***n*** <=53243(CFFBhex)

Note: Range given is RAM based position recording point memory. If not using the PR function, then this memory is available for general use.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Related Commands: **GB, GL, GW, PB, PL**

RA*n*

Copy register to accumulator

This command copies user register ***n*** to the accumulator. RA***n*** is interpreted as follows:

accumulator = register ***n***

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine divides 7 by 2.

AR50,+          ;Save accumulator to reg. 50.
AL0,AR1,+       ;Initialized high order bits to 0.
AL7,+           ;Initialize low order bits to "7".
AD2,+           ;Divide by "2".
AR10,+          ;Save result in user register 10.
RA50+           ;Restore the accumulator.
```

Range: 1 <= ***n*** <= 255

Registers: User Register 0, accumulator
User Register ***n***

The contents of the accumulator may be viewed by executing TR0<*CR*>. The contents of register ***n*** may be viewed by executing TR***n***<*CR*>.

Related Commands: **AR, AL**

RB*n*

Servo table byte to accumulator

This command copies a byte located at servo table offset ***n*** to the accumulator. RB***n*** is interpreted as follows:

accumulator = servo table byte @ offset ***n***

Range: See appendix D, Servo Table Offset List.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine reports the present FR setting.

RB110,+     ;Read derivative sample period.
TR0+        ;Tell accumulator value.
```

Related Commands: RW, RL, WB, WW, WL

RL*n*

Servo table long word to accumulator

This command copies a long word located at servo table offset ***n*** to the accumulator. RL***n*** is interpreted as follows:

accumulator = servo table long word @ offset ***n***

Range: See appendix D, Servo Table Offset List.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine reports the auxiliary encoder
;       input position.

RL54,+      ;Read Aux. Enc. input position.
TR0+        ;Tell accumulator value.
```

Related Commands: RB, RW, WB, WW, WL

RW*n*

Servo table word to accumulator

This command copies a word located at servo table offset ***n*** to the accumulator. RW***n*** is interpreted as follows:

accumulator = servo table word @ offset ***n***

Range: See appendix D, Servo Table Offset List.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine reports the torque limit setting.

RW96,+          ;Read torque limit setting.
TR0+            ;Tell accumulator value.
```

Related Commands: **RB, RL, WB, WW, WL**

SL*n*

Accumulator shift left

This command shifts the accumulator value left by ***n*** bits. Most significant ***n*** bits are lost. Least significant ***n*** bits are set to zero.

Range: 0 <= ***n*** <= 31

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine demonstrates the SL command.

AL1,+           ;Load accumulator with value 1.
SL1,+           ;Shift left by one bit.
TR0+            ;Tell accumulator value.
```

Related Commands: **SR**

SR*n*

Accumulator shift right

This command shifts the accumulator value right by ***n*** bits. The least significant ***n*** bits are lost. The most significant ***n*** bits are set to zero.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: 0 <= ***n*** <= 31

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Example:

```
;       This routine demonstrates the SR command.

AL3,+       ;Load accumulator with value 1.
SR1,+       ;Shift left by one bit.
TR0+        ;Tell accumulator value.
```

Related Commands: **SL**

TR*n*

Tell register

This command reports the value located in user register ***n***.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: 0 <= ***n*** <= 255

Registers: User Register ***n***

User register ***n*** may be viewed by executing TR***n***<*CR*>.

Example:

```
;       This is a command line example where TR
;       is used to verify the value in register 10.

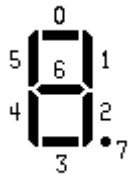
>AL5,AR10,TR10    ;<CR>
0000000005        ;A value +<CR> is returned,
>                 ;and then a new prompt.
```

Related Commands: **DM, HM**

UD*n*

Update display

This command writes the value ***n*** to the seven segment status display. Status display segments are arranged by bit number as follows:



Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine writes an "F" to the display.

UD113+          ;Turn bits 0, 4, 5 and 6 on.
```

Range: 0 <= ***n*** <= 255

Registers: none.

Related Commands: **GN, GS**

WB*n*

Accumulator to servo table byte

This command copies the accumulator to a byte located at servo table offset ***n***. WB***n*** is interpreted as follows:

servo table byte @ offset ***n*** = accumulator

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Warning !: Writing to the servo table is not recommended. Unpredictable motion may occur. Dedicated parameter setup commands, such as SA, SV and DS, exist for writing to most of the table values. Consult Automotion for assistance if use this command is required.

Range: See appendix D, Servo Table Offset List.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Related Commands: **RB, RW, RL, WW, WL**

WL*n*

Accumulator to servo table long word

This command copies the accumulator to a long word located at servo table offset ***n***. WL***n*** is interpreted as follows:

servo table long word @ offset ***n*** = accumulator

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Warning !: Writing to the servo table is not recommended. Unpredictable motion may occur. Dedicated parameter setup commands, such as SA, SV and DS, exist for writing to most of the table values. Consult Automotion for assistance if use this command is required.

Range: See appendix D, Servo Table Offset List.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Related Commands: **RB, RW, RL, WB, WW**

WW*n*

Accumulator to servo table word

This command copies the accumulator to a word located at servo table offset ***n***. WW***n*** is interpreted as follows:

servo table word @ offset ***n*** = accumulator

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Warning !: Writing to the servo table is not recommended. Unpredictable motion may occur. Dedicated parameter setup commands, such as SA, SV and DS, exist for writing to most of the table values. Consult Automotion for assistance if use this command is required.

Range: See appendix D, Servo Table Offset List.

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Related Commands: **RB, RW, RL, WB, WL**

BK

Break

Execution of this command will cause the rest of the command string or macro string to be skipped. This command can be used with sequence commands to implement conditional execution.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine checks for all fault bits in the
;       Servo Status register.

RL0,+           ;Servo Status to accumulator.
AN16711680,+    ;Accumulator AND FF0000 hex.
IG0,+           ;If bits 16 to 23 are set do next 2.
TR0,+           ;Report fault code from status.
BK,+            ;Break out of program.
UD0+            ;Else, clear status display & cont.
```

Range: none.

Registers: none.

Related Commands: **IB, IE, IG, IU**

BR*n*

Baud rate

This command selects the RS232 or RS485 serial baud rate based on the code ***n***. When BR is executed, the DSC will immediately switch to the new baud rate setting. The default baud rate is ***n*** = 51, or 9600 baud.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range:

n	=	25 :	19200 baud
	=	51 :	9600 baud
	=	103 :	4800 baud
	=	207 :	2400 baud
	=	416 :	1200 baud
	=	832 :	600 baud

Registers: none.

Related Commands: **EF, EN, PC, XF, XN**

CA*n*

Convert register to ASCII

This command converts a signed integer value, located in user register ***n***, to an ASCII code string. The resulting 11 byte string, "+" or "-" and 10 numerical characters, is stored at the physical address value returned in the accumulator.

Range: 0 <= ***n*** <= 255

Registers: User Register 0, accumulator

The contents of the accumulator may be viewed by executing TR0<*CR*>.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

```
;       This routine demonstrates the CA command.

AL0123456789,+  ;Load accumulator with value.
AR10,+          ;Place value in register 10.
CA10,TR0+       ;Convert 10 and tell address.

;       This routine returns the ASCII code string.

AR20,GB@0,+     ;Get byte at address in acc.
TR0,RA20,+      ;Tell acc, and retrieve address
AA1,RP10+       ;Index address and loop.
```

Related Commands: **GB**

CE*n*

Command error enable

This command determines how the DSC will respond to the occurrence of a command interpreter error. CE1 will cause the DSC to disable the servo and write a "C." to the status display when a command interpreter error is detected. CE0 will leave the servo enabled and status display undisturbed. In either case however, the DSC will abort any pending command execution, transmit a "?-<*error code*> <*CR*> <*prompt*>" string, and wait for a new command.

Range: ***n*** = 0 : disable
= 1 : enable

Registers: Servo Status, bit 22

Bit 22 is "command error" detected. This condition is cleared by execution of MF or MN.

Usage:
Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Command Interpreter Error Codes:

?-1	Parameter error.
?-2	Command not valid.
?-3	Negative repeat.
?-4	Macro definition command not first.
?-5	Macro number out of range.
?-6	Macro does not exist.
?-7	Command canceled.
?-8	Axis number out of range.
?-9	Motor on not allowed during command.

Related Commands: **MF, MN, RT, TE**

DM

Decimal mode

This command configures the DSC to transmit, receive and interpret values as decimal numbers. Unless otherwise indicated, all examples given in the DSC command Summary assume that the DSC is operating in decimal mode.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: none.

Registers: none.

Related Commands: **HM**

EF

Echo off

This command disables character echoing through the RS232 or RS485 serial port. Only data requested by the host will be transmitted by the DSC. This mode of communication is normally used when the host is operating in half duplex.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: none.

Registers: none.

Related Commands: **EN, XF, XN**

EN

Echo on

This command enables character echoing through the RS232 or RS485 serial port. Data transmitted by the host will be echoed by the DSC. This mode of communication is normally used when the host is operating in full duplex.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: none.

Registers: none.

Related Commands: **EF, XF, XN**

HM

Hex mode

This command configures the DSC to transmit, receive and interpret values as hexadecimal numbers. Hexadecimal numbers are character case sensitive. Capital A through F are required.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

; This is a command line example.

```
>HM,TS,DM     ;Hex mode,TS,Dec. mode<CR>
0080422F      ;Hex value+<CR> is returned,
>             ;and then a new prompt.
```

Range: none.

Registers: none.

Related Commands: **DM**

NO

No operation

This command performs no operation. This command can be used as fill for command string branching. See related commands.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: none.

Registers: none.

Related Commands: **IB, IE, IG, IU**

PC*n*

Prompt character

This command configures the DSC prompt character. The value ***n*** is the ASCII code for the prompt character to be used.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

; This is a command line example.

>PC35 **;**Prompt = "#"<*CR*>
**;**A new prompt is returned.

Range: 0 <= ***n*** <= 127

Registers: none.

Related Commands: **DM, HM**

RT

Reset MPU

This command performs a complete restart of the DSC firmware, including restoration of all default conditions such as acceleration and velocity settings, and leaves the servo in the "off" state.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Range: none.

Registers: All.

Related Commands: **IB, IE, IG, IU**

XF*n*

X-off code

This command sets an 8 bit code that is transmitted from the DSC to the host when the DSC begins executing a command, or command string, received from the host.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

; This is a command line example.

```
>XN35,XF36      ;Xon = "#", Xoff = "$" <CR>.
#>TP            ;Xon+">" is received, TP<CR>.
$0000000000     ;Xoff + <position> +
#>              ; Xon +">" is received.
```

Range: 0 <= ***n*** <= 255

Registers: none.

Related Commands: **BR, EF, EN, PC, XN**

XN*n*

X-on code

This command sets an 8 bit code that is transmitted from the DSC to the host when the DSC completes exicution of a command, or command string, received from the host.

Usage:

Position Mode - Yes
Velocity Mode - Yes
Gain Mode - Yes
Torque Mode - Yes
Slave Mode - Yes
Contour Mode - Yes

Example:

; This is a command line example.

```
>XN35,XF36      ;Xon = "#", Xoff = "$" <CR>.
#>TP            ;Xon+">" is received, TP<CR>.
$0000000000     ;Xoff + <position> +
#>              ; Xon +">" is received.
```

Range: 0 <= ***n*** <= 255

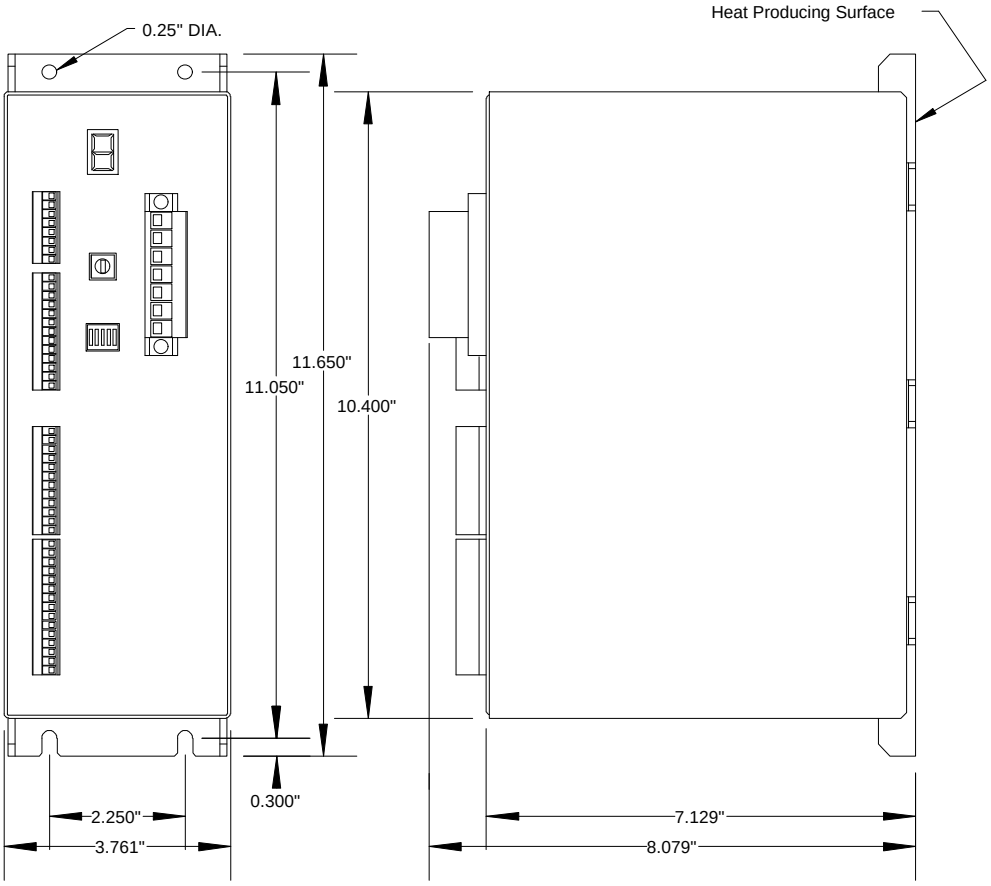
Registers: none.

Related Commands: **BR, EF, EN, PC, XF**

Section 6.0

DSC Hardware Reference

DSC2000 Dimensions



Note: **1.** All dimensions given are in inches.

Mounting

! Important:

1. Read all safety precautions in Section 1.0 before installing the DSC.

2. The DSC is designed to panel mount inside a NEMA type enclosure. Enclosure should be filtered and ventilated, or cooled as required to limit internal ambient temperature to less than 40 degrees Celsius.

3. Reference DSC Package Outline on page 6-1. The DSC internally monitors the temperature of the "Heat Producing Surface" and will automatically produce an amplifier fault (Status: "A.") if this surface reaches 50 to 55 degrees Celsius. Always mount the DSC to a suitable cold plate surface. Optional factory mounted heatsinks are available from Automotion.

4. The DSC is designed to mount vertical with ventilation hole patterns top and bottom for optimal convection cooling.

Line Power Requirements

1. The DSC rectifies and filters power from the user supplied AC line. Line power service should be capable of supplying at least 1.5 times the DSC peak output phase current rating while maintaining Line Voltage requirements. See general specifications or DSC Name Plate ratings for additional information.

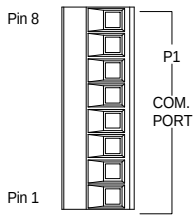
Line Power Conditioning

! Important:

1. Fusing internal to the DSC is supplemental only and not user serviceable. Branch fuse protection external to the DSC must be specified by the user per applicable end use Code or Standard. Contact Automotion if assistance is required.

2. Automotion recommends using an EMI line filter to condition power to the DSC. This filter should be rated for the AC line voltage (115 or 230) and rated for at least 1.5 times the DSC peak output phase current rating.

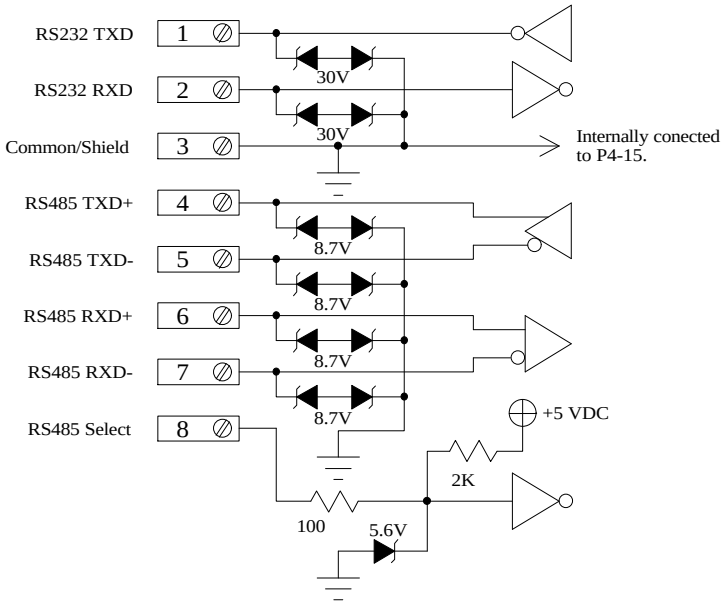
3. If using a power disconnect device to routinely interrupt power to the DSC, consideration should be given to AC line inrush current. When fully discharged, the DSC can draw up to 200 Amps for 1 cycle of the AC line each time AC power applied.



Pin Description

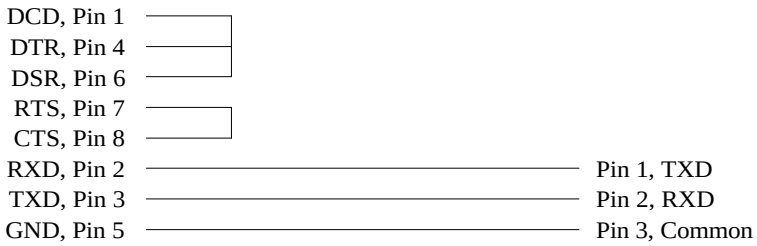
Pin	Name	I/O	Specifications
1	RS232 TXD	Out	Line driver, 1488 equivalent.
2	RS232 RXD	In	Line receiver, 1489 equivalent.
3	Common	-	Internally connected to P4-15.
4	RS485 TXD+	Out	Transceiver, 75176 equivalent.
5	RS485 TXD-	Out	Transceiver, 75176 equivalent.
6	RS485 RXD+	In	Transceiver, 75176 equivalent.
7	RS485RXD-	In	Transceiver, 75176 equivalent.
8	RS485 Select	In	Active low, tie to P1-3 for RS485.

Equivalent Circuit

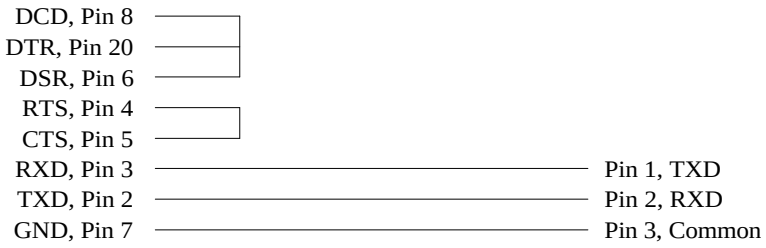


! Important: The maximum length of an RS232 cable should not exceed 50 feet. The maximum length of an RS422 or RS485 cable should not exceed 4000 feet. For best performance, Automotion recommends Beldon 9830 (RS422), 9842 (RS485) or equivalent cable.

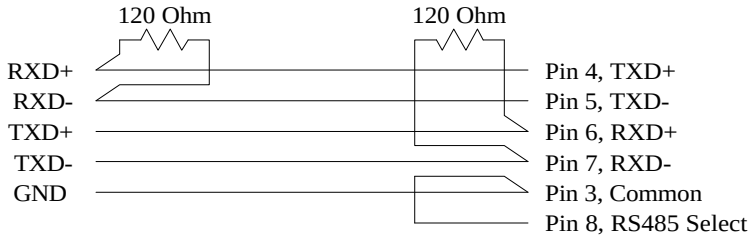
Typical PC, 9 Pin, RS232 to P1



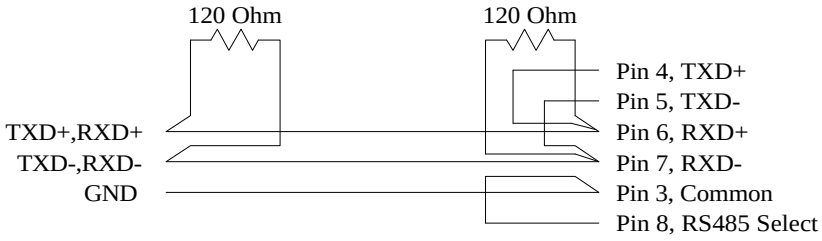
Typical PC, 25 Pin, RS232 to P1



Typical RS422 to P1

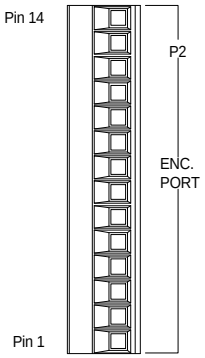


Typical RS485 to P1



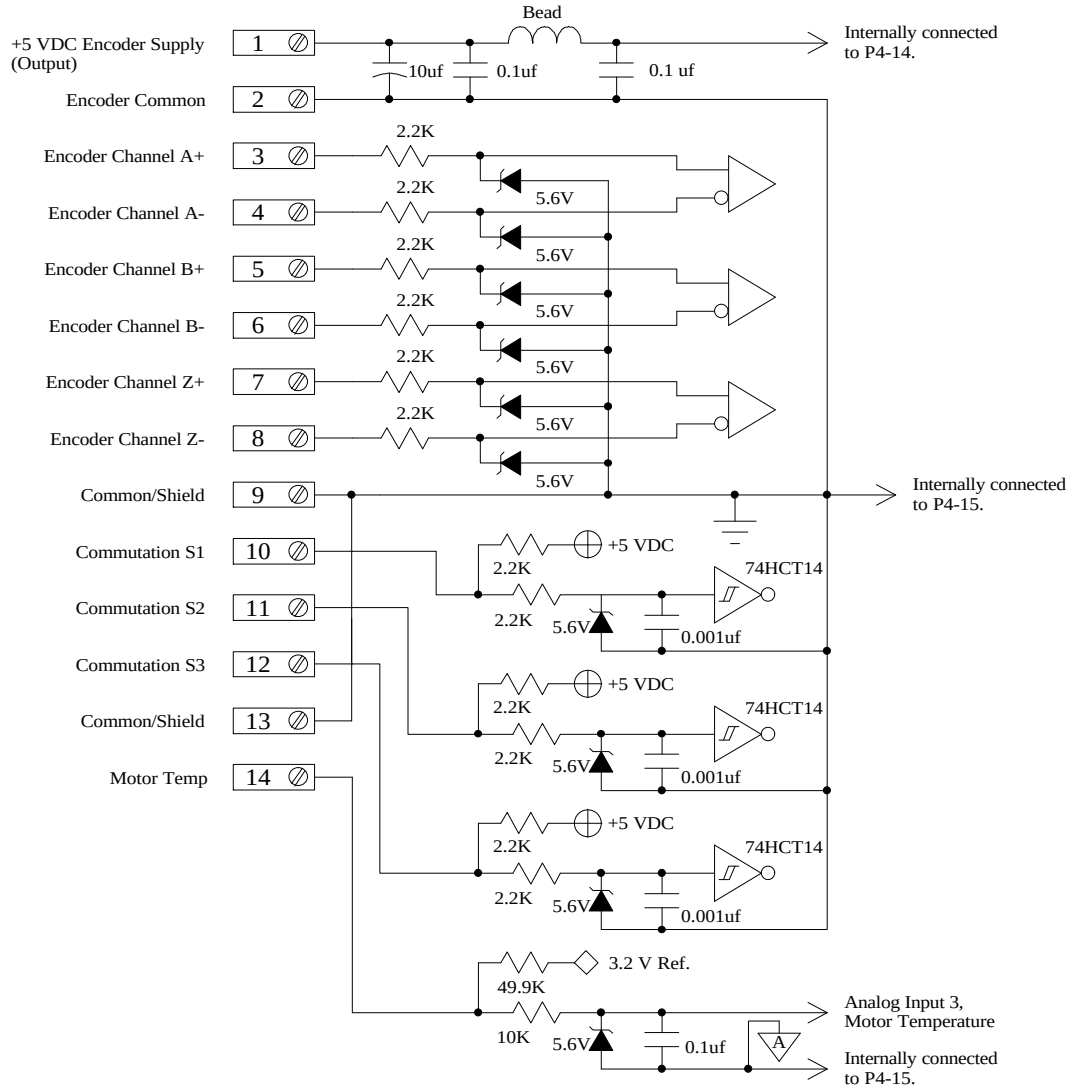
Note: 120 Ohm terminating resisters are required for RS422 or RS485. If multiple DSC's are connected, then resisters should be placed at each end of the network and not at every node.

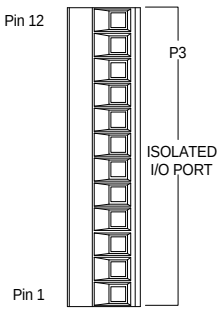
Pin Description



Pin	Name	I/O	Specifications
1	Enc. Supply	Out	Filtered, internally connected to P4-14.
2	Enc. Common	-	Internally connected to P4-15.
3,4	Enc., A+,A-	In	422 line receiver, 26LS32 equivalent.
5,6	Enc., B+,B-	In	422 line receiver, 26LS32 equivalent.
7,8	Enc., Z+,Z-	In	422 line receiver, 26LS32 equivalent.
9	Enc. Common	-	Internally connected to P4-15.
10,	Comm. S1	In	TTL compatible, 74HCT14 or equivalent.
11	Comm. S2	In	TTL compatible, 74HCT14 or equivalent.
12	Comm. S3	In	TTL compatible, 74HCT14 or equivalent.
13	Enc. Common	-	Internally connected to P4-15.
14	Motor Temp.	In	Analog, 0 to 3.2Vdc.

Equivalent Circuit

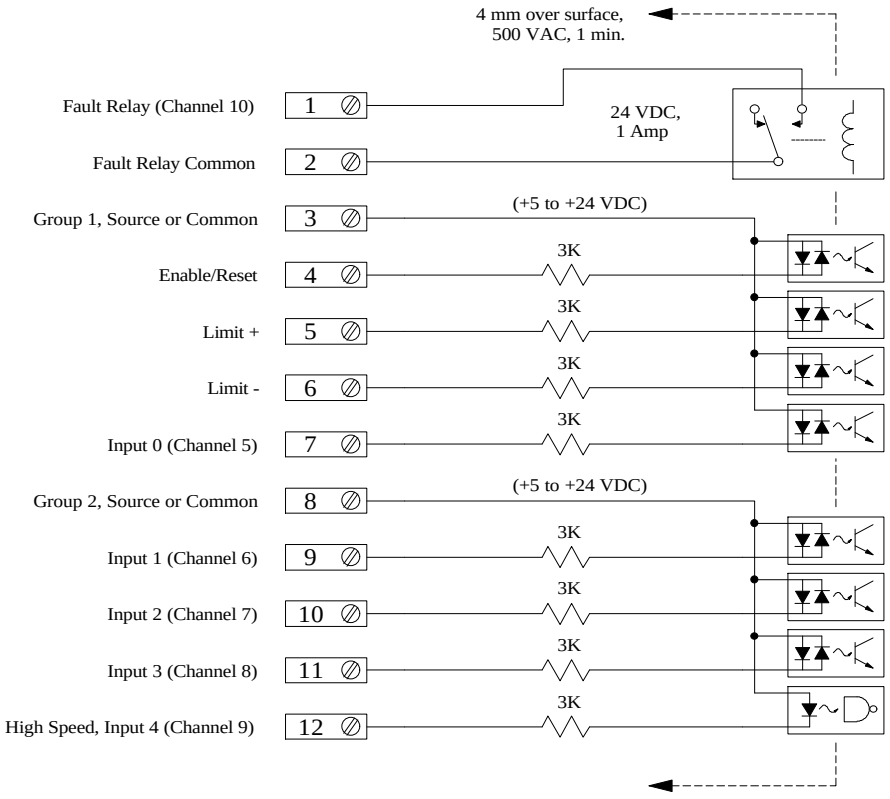




Pin Description

Pin	Name	I/O	Specifications
1	Fault Relay	Out	24 Vdc, 1 Amp
2	Fault Common	-	Isolated, 500 Vac.
3	Group 1 S/C	-	Group 1, P3-4 to P3-7, Source / Common.
4	Enable/Reset	In	5 to 24 Vdc, minimum ON current: 1.25 mA
5	Limit+	In	5 to 24 Vdc, minimum ON current: 1.25 mA
6	Limit-	In	5 to 24 Vdc, minimum ON current: 1.25 mA
7	In 0 (Chn. 5)	In	5 to 24 Vdc, minimum ON current: 1.25 mA
8	Group 2 S/C	-	Group 2, P3-9 to P3-12, Source / Common.
9	In 1 (Chn. 6)	In	5 to 24 Vdc, minimum ON current: 1.25 mA
10	In 2 (Chn. 7)	In	5 to 24 Vdc, minimum ON current: 1.25 mA
11	In 3 (Chn. 8)	In	5 to 24 Vdc, minimum ON current: 1.25 mA
12	In 4 (Chn. 9)	In	5 to 24 Vdc, High Speed, open collector input only. Minimum on current = 1.5 mA

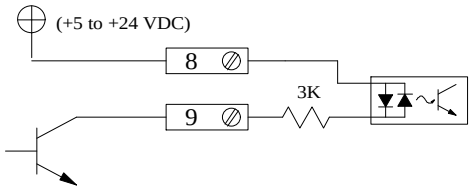
Equivalent Circuit



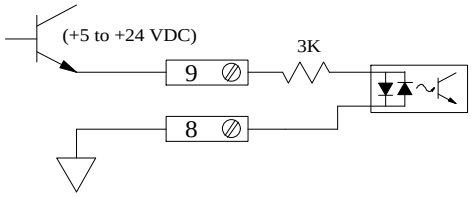
Note: The Fault Relay output is DSC I/O channel number 10. This output is controlled using channel type commands: CN, CF, etc. This output can only be energized (i.e. contact closed) if DSC fault conditions are clear. The Fault Relay will automatically open when the DSC detects certain fault conditions:

Status Display = "b.", "P.", "r.", "F.", "t.", "A.", "C."

Typical Open Collector Interface



Typical Emitter Follower Interface

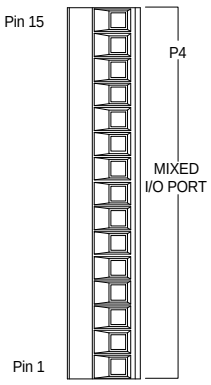


Note:

1. Dry contact devices may be substituted for solid state devices shown above. See Channel Commands in section 5 for information on reading inputs 0 to 4.

2. Enable/Reset is a dedicated input. The DSC is held in a Reset state when no current is flowing into, or out of terminal P3-4. See appendix D for additional information.

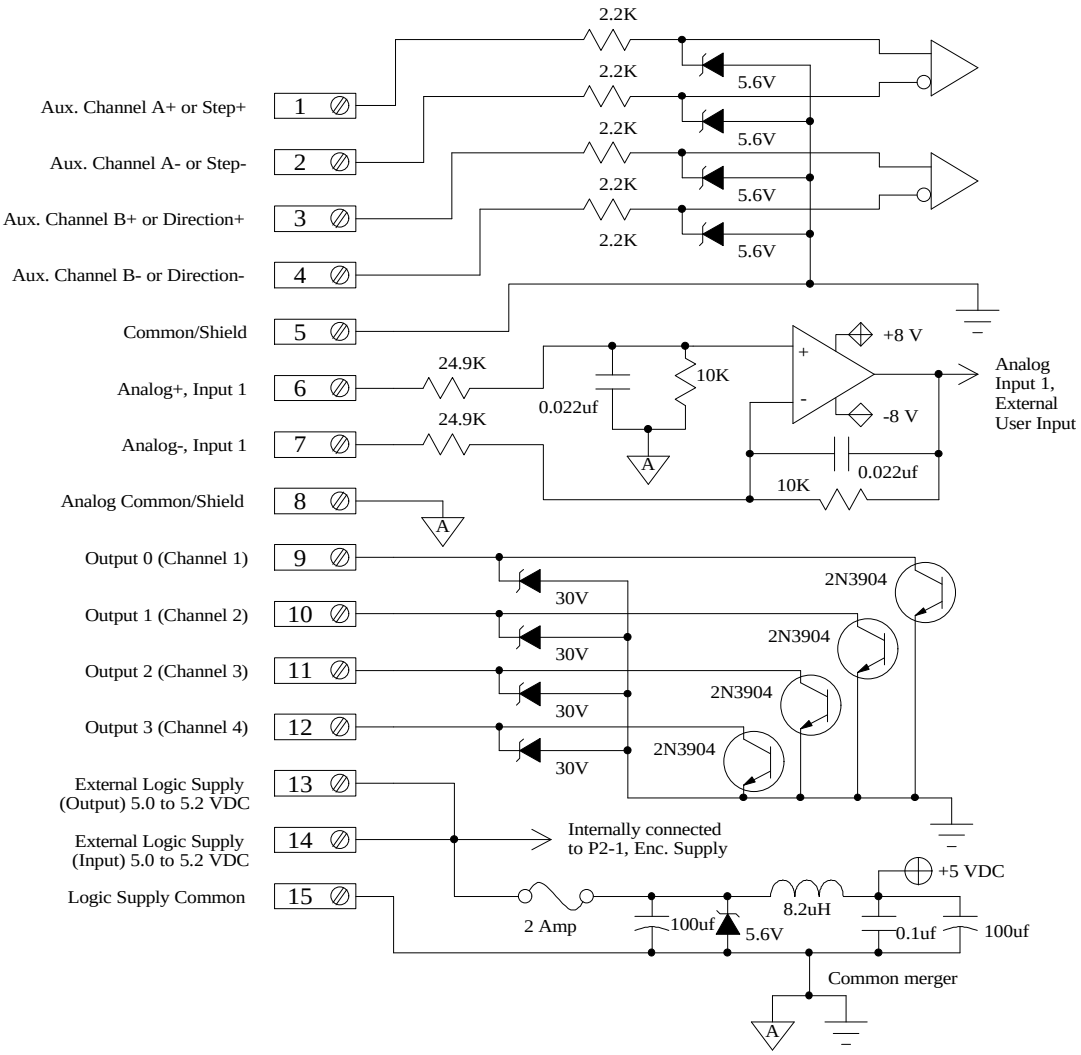
3. Limit+ and Limit- are dedicated inputs when enabled. See Parameter Setup commands LM, LN, and LF in Section 5 for additional information. If not enabled, Limit+ and Limit- can be read as bits 31 and 28 in the DSC Status register.



Pin Description

Pin	Name	I/O	Specifications
1,2	Aux A/Step	In	422 line receiver, 26LS32 equivalent.
3,4	Aux B/Dir.	In	422 line receiver, 26LS32 equivalent.
5	Common	-	Internally connected to P4-15
6,7	Analog In 1	In	+/-10 Vdc differential, Analog input 1.
8	Analog Com	-	Internally connected to P4-15
9	Out 0 (Chn. 1)	Out	5-24 Vdc, NPN open collector, 0.5V@20mA
10	Out 1 (Chn. 2)	Out	5-24 Vdc, NPN open collector, 0.5V@20mA
11	Out 2 (Chn. 3)	Out	5-24 Vdc, NPN open collector, 0.5V@20mA
12	Out 3 (Chn. 4)	Out	5-24 Vdc, NPN open collector, 0.5V@20mA
13	Ext. Logic Pwr	Out	5.0 Vdc, Internally connected to P4-14.
14	Ext. Logic Pwr	In	5.0 - 5.2 Vdc, user supplied, See Note 1.
15	Ext. Common	-	User supplied, single point DSC logic common.

Equivalent Circuit



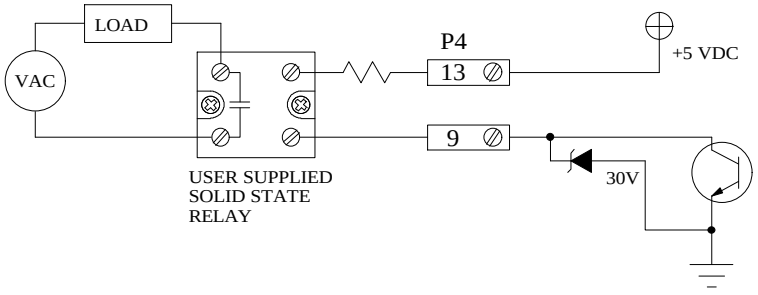
Note: **1.** External logic supply current draw can be calculated as follows:

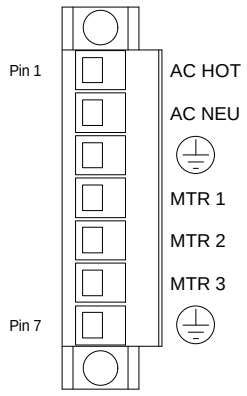
Total Amp = [0.75 + (P4-13 load) +(P2-1 load)] * (safety factor)
If using Automotion SP motor: (P2-1 load) = 250 mA

2. Internal fuse is not user serviceable. If damaged, contact Automotion for service and repair.

3. Auxiliary Encoder or Step and Direction input is user programmable. See Slave Mode operation for addition information. 2 MHz edge rate (500 KHz Pulse frequency) maximum.

Typical Output Interface, AC Switching

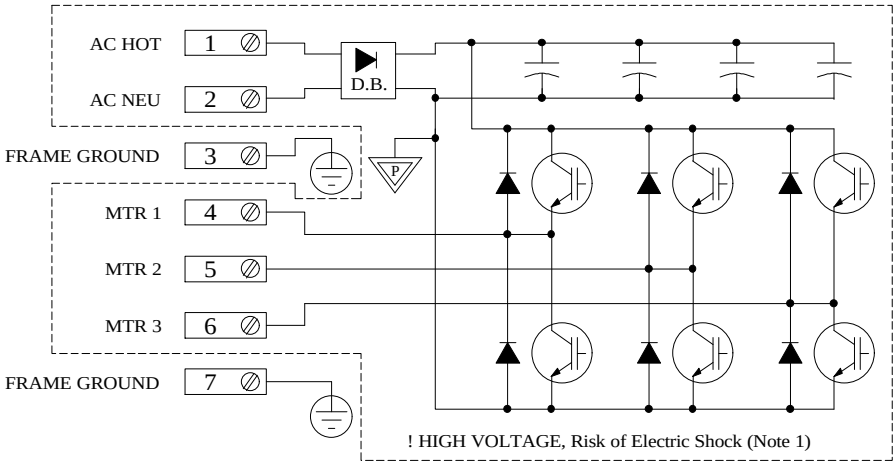




Pin Description

Pin	Name	I/O	Specifications
1	AC HOT	In	DSCxx0x-: 90-132 Vac, 50/60 Hz. DSCxxDx-: 180-264 Vac, 50/60 Hz. (Internally fused, not user serviceable)
2	AC NEU	In	DSCxx0x-: 90-132 Vac, 50/60 Hz. DSCxxDx-: 180-264 Vac, 50/60 Hz.
3	FRAME GND	-	Internally connected to DSC chassis.
4	MTR 1	Out	DSCxx0x-: 160 Vdc nominal, 18 KHz PWM DSCxxDx-: 320 Vdc nominal, 18 KHz PWM
5	MTR 2	Out	DSCxx0x-: 160 Vdc nominal, 18 KHz PWM DSCxxDx-: 320 Vdc nominal, 18 KHz PWM
6	MTR 3	Out	DSCxx0x-: 160 Vdc nominal, 18 KHz PWM DSCxxDx-: 320 Vdc nominal, 18 KHz PWM
7	FRAME GND	-	Internally connected to DSC chassis.

Equivalent Circuit



! Important:

1. AC HOT, AC NEU, MTR 1, MTR 2 and MTR 3 connections are not isolated from the user supplied AC power source. These power circuits are "floating" on the AC Line.

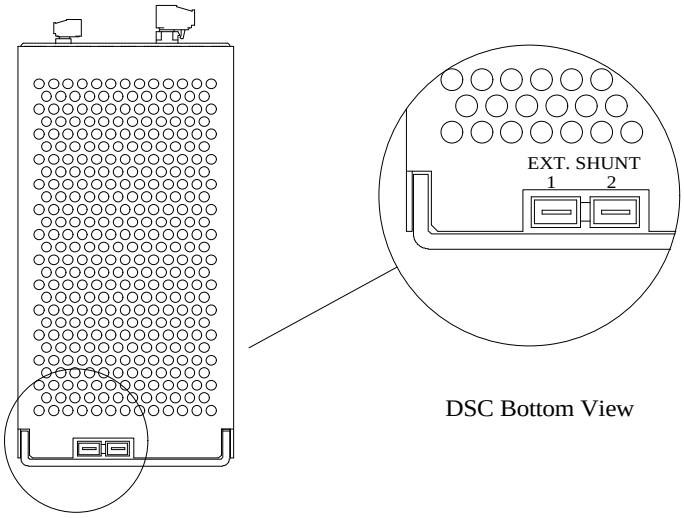
2. Galvanic isolation between "AC HOT, AC NEU and associated power circuits" to "FRAME GROUND" is provided: 1500 VAC, 1 min. HI-POT, 4 mm over surface typical.

3. Galvanic isolation between "AC HOT, AC NEU and associated power circuits" to "Logic Common (P4-15) and associated circuits" is provided: 1500 VAC, 1 min. HI-POT, 8 mm over surface typical.

! Important: Read this Section carefully before installing an external shunt resistor. Improper installation or application of an external shunt can result in serious personal injury, fire and electric shock.

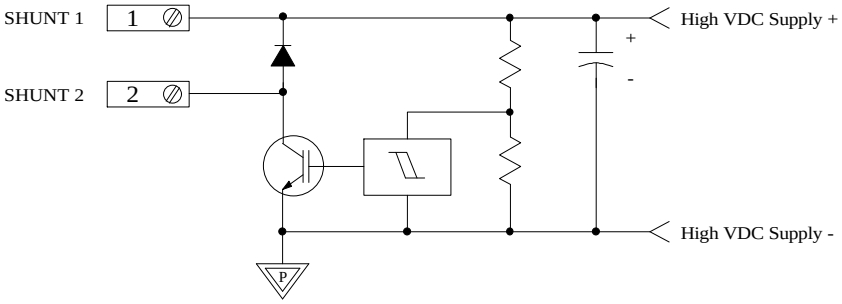
The following applies to DSC model numbers: DSCxxxx-xxx-2xxx

Pin Location



DSC Bottom View

Equivalent Circuit



! Important:

1. SHUNT 1 and SHUNT 2 connections are not isolated from the user supplied AC power source. These power circuits are "floating" on the AC Line.

2. Galvanic isolation between "SHUNT 1, SHUNT 2 and associated power circuits" to "FRAME GROUND" is provided: 1500 VAC, 1 min. HI-POT, 4 mm over surface typical. To maintain this level of isolation, proper shunt selection and installation is required.

External Shunt Operation

During rapid deceleration of the motor, power regeneration occurs. This causes kinetic energy to be returned to the DSC control. For small inertial loads this may not be a problem, but for larger loads this returned energy must be dissipated by the control. The amount of energy return will depend upon both the rate of motor deceleration plus the size of the motor load.

The DSC control has an automatic means of dissipating this "surplus" returned energy. It is done by the use of what is called a shunt regulator within the control. To function correctly the user must connect an appropriately sized external shunt resistor across pins SHUNT 1 and SHUNT 2.

If no external shunt resistor is installed the user must understand that excess regeneration may cause nuisance trips. While this occurrence will not damage the control, it will disrupt the normal operation of the motor.

Selection of an External Shunt Resister

The duty cycle for an external shunt resistor is application dependent. As such it helps to estimate just how much kinetic energy is stored in the motor and its load when deceleration begins. It is this kinetic energy which transfers back to the drive whenever a deceleration cycle begins. To estimate the energy rate that is returned to the drive during a deceleration cycle use the following formula:

Average Regen Power =	{(Motor RPM's at start of decel cycle) + (Motor RPM's at end of decel cycle)} x (Motor decel torque) x (0.05236)
Peak Regen Power =	(Motor RPM's at start of decel cycle) x (Motor decel torque) x (0.10472)
Where:	Both Average Regen Power and Peak Regen Power are in units of Watts, Motor decel torque is in units of Newton meters.
Note:	Newton meters is calculated by multiplying the average deceleration current level by the motor's torque constant, Kt (Nm/A)

In general it is desirable that the Peak Regen Power be less than 720 Watts for 115 VAC sourced controls, and less than 2880 Watts for 230 VAC sourced controls. The minimum shunt resistance permitted for all DSC models is 50 Ohms.

Automotion DOES NOT recommend the use of a shunt resistor with less than a 50 Watt rating for any application. Consult Automotion if you wish to violate these guidelines.

Automotion recommends that the resistor type be a wire wound, low inductance, ceramic core, all welded construction. The Ohmite series "250 type' is a possibility.

The final choice for a resistor is dependent upon the Peak and Average Regen Power, the total deceleration time and how often the decel cycle is repeated. The Ohmite product listed above is designed to handle momentary over stress of up to 5 times the rated wattage for 5 seconds. Tests at Automotion indicate that the over stress can be up to 12 times the rated wattage on the 55 Watt size Ohmite part for 2 seconds, provided the cycle is not repeated very often.

In general it is desirable to keep the surface temperature of the shunt resistor below 60° C under worst case conditions. If this cannot be done with a single shunt resistor, then multiple resistors may need to be used to distribute the stress across more parts. For example, two 100 Ohm/55 Watt resistors could be electrically wired in parallel. This would still meet the 50 Ohm minimum for a 115 VAC sourced drive yet it would double the wattage capacity.

It is also important to use an ohmic value as large as possible. This may be approximated using the following formulas:

For a 115 VAC sourced drive:

Minimum ohmic value = {(40000) / (Peak Regen Power*)} x {0.9}
*Must be < 720 Watts.

For a 230 VAC sourced drive:

Minimum ohmic value = {(160000) / (Peak Regen Power*)} x {0.9}
*Must be < 2880 Watts.

Location and Mounting of the External Shunt Resister

The shunt resistor is a heat producing element powered from high voltage electricity. Consequently the user must follow several common sense safety guidelines to avoid possible problems later. The following is a list of suggested methods for mounting the external shunt resistor:

! Important:

1. DO NOT mount the shunt in the proximity of flammable materials. It is suggested that it be mounted only upon a metal surface, such as steel. It is also suggested that if it is to be installed in an enclosure, then this enclosure be made out of nonflammable metal materials with flow through air ventilation.

2. DO NOT use the shunt in an explosive atmosphere.

3. DO NOT use the shunt in a location where it may be exposed to humidity condensation or other liquids.

4. Always locate the shunt in a location where it will remain clean and dry.

5. The shunt is activated by potentially lethal voltages. Mount the shunt someplace where it will not expose the user or service personnel to a shock hazard. A 115 VAC sourced drive can deliver voltages between 200 and 250 volts to the shunt. A 208/230 VAC sourced drive can deliver voltages between 400 and 450 volts to the shunt.

6. Always locate the shunt resistor in a location where it has space for ventilation. DO NOT locate in a closed space without free air movement.

The following formulas are an indication of the maximum continuous wattage developed across the shunt whenever it is ON. We suggest you design your installation to be fire and shock safe with this information in mind.

For a 115 VAC sourced drive:

Maximum Wattage = (34150) / (Shunt Resistance in Ohms)

For a 230 VAC sourced drive:

Maximum Wattage = (132500) / (Shunt Resistance in Ohms)

Temperature limiting devices are available to aid in the removal of the power source from the shunt resister in the event it begins to overheat for any reason. However, they are not a substitute for making the basic installation fire proof.

The simplest way to limit the heat concentration at the shunt resistor is to use the highest ohmic value resister(s) possible plus higher wattage capacity in every application. This will help to keep the shunt resister surface temperature as low as possible under all circumstances. Worst case testing is always required to insure safe performance at all times.

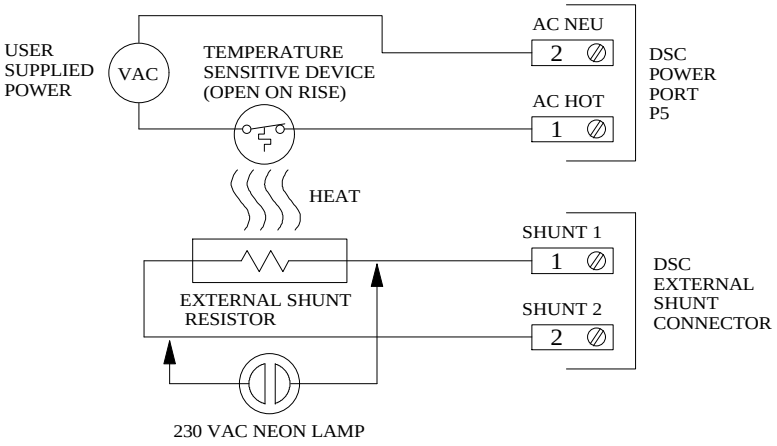
Suggested Wiring Methods for the External Shunt Resistor

At the user's option, a temperature sensitive fuse or switch can be used in conjunction with the external shunt resistor. The function of this device is to limit the temperature rise of the shunt resistor should it ever overheat. The following devices have been tried by Automotion and show merit in this purpose.

1. Elmwood part #JD090-V231.
2. Therm-O-Disc part #G5A01077C.
3. RCD Components fusible resistor #TF04D/.25 Ohm/±5% Tol.
4. Bussman heat limiter fuse part #TGC.

Devices 1 and 2 are suitable for application as shown in Figure 1 ONLY! All of these listed devices are suitable for application as shown in Figure 2. Each part has a slightly different threshold temperature for opening. Check the manufacturers data sheet carefully before use.

Figure 1

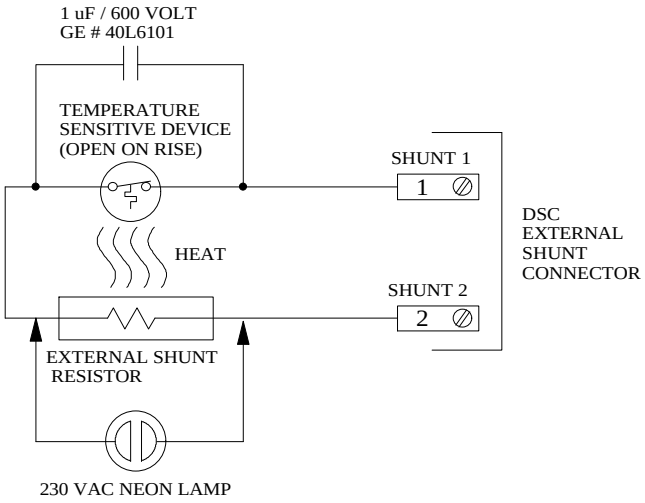


The optional neon lamp is intended to show the user just how heavy a duty cycle is applied to the shunt resistor. It will blink every time the shunt resistor is toggled. If it blinks with a very short duty cycle under worst case deceleration conditions, the chances are that a higher ohmic resistor value could be used. If it DOES NOT blink at all, then an external shunt resistor may not be needed.

If the user decides to use a temperature sensitive fuse or switch, it is very important that this device be located in close proximity to the shunt resister. It is obvious that if the surface temperature of the shunt resistor becomes excessively hot, the components listed on the previous page must respond effectively. Generally the

temperature sensitive device should be mounted in contact with the surface of the shunt resistor or very close to it.

Figure 2



Automotion favors the use of the installation shown in Figure 1 whenever possible. This is because it requires fewer components. If the installation in Figure 2 is used, the 1 uf capacitor should be located away from the path of heat if the shunt resistor should over heat.

The capacitor shown in figure 2 is made by General Electric under their part #40L6101. It may be possible to find an equivalent substitute. It is non-polarized and rated for 600 Volt DC service. If the installation shown in Figure 2 is used then this capacitor is required. Any installation without it is likely to render the temperature sensing device ineffective.

Whichever method is chosen to locate the external shunt resistor, it should be wired to the Automotion control using at least 16 AWG wire rated for 600 VAC service. Automotion recommends using wire that is jacketed with insulating material sufficiently rated for flammability and high temperature. It is important that the wire connecting the shunt resistor is properly dressed using good installation practice to prevent any short circuits from happening. Short circuits will damage the Automotion control.

The final method of installation should be tested to confirm that it functions correctly and safely. Also, apply heat to the temperature sensing device chosen to confirm that it works consistently in the end use environment. Multiple test are recommended.

Any questions regarding the external shunt resistor should be directed to Automotion. These recommendations are intended to provide some help in selection and installation of an external shunt resistor. They are not a substitute for using good judgment and following recognized standards in the end use industry for safe and reliable installation.

Appendices

Alphabetical Command Reference ..... A
Firmware Default Settings, Servo Table ..... B
Firmware Revision History C
Fault Codes, Operation and Recovery ..... D
DSC to SP Motor Switch Settings ..... E
DSC Commutation Sequence Tables ..... F
Suggested method for current loop tuning ..... G

AAn	Accumulator add	5-58
AB	Abort motion	5-17
AC	Accumulator compliment	5-58
ADn	Accumulator divide	5-59
AEn	Accumulator exclusive OR	5-59
AGn	Acceleration gain	5-5
ALn	Accumulator load	5-60
AMn	Accumulator multiply	5-60
ANn	Accumulator AND	5-61
AOn	Accumulator OR	5-61
ARn	Copy accumulator to register	5-62
ASn	Accumulator subtract	5-62
AT	Tell auxiliary position	5-26
BCn	Breakpoint channel	5-43
BK	Break	5-73
BRn	Baud rate	5-73
CAn	Convert register to ASCII	5-74
CEn	Command error enable	5-74
CFn	Channel off	5-43
CHn	Channel high true logic	5-44
CLn	Channel low true logic	5-44
CMn	Contour mode	5-17
CNn	Channel on	5-45
DBn	Target deadband	5-5
DDn	Direction Deadband	5-6
DFn	Do if channel off	5-46
DGn	Deceleration Gain	5-6
DHn	Define home	5-18
DIn	Velocity direction	5-18
DM	Decimal mode	5-75
DNn	Do if channel on	5-46
DOn	Display optimal position	5-26
DQn	Display recorded torque	5-27
DRn	Display recorded position	5-27
DSn	Deceleration Set	5-7
DTn	Delay at target	5-47
DYn	Set record delay	5-28
EF	Echo off	5-75
EN	Echo on	5-76
FIn	Find index	5-19
FRn	Set derivative sample period	5-7
GAn	Copy analog input to accumulator	5-63
GBn	Copy byte to accumulator	5-63
GH	Go home	5-19
GLn	Copy long word to accumulator	5-64
GM	Gain mode	5-20
GN	Copy node switch to accumulator	5-64
GO	Start velocity	5-20
GS	Copy select switch to accumulator	5-65
GWn	Copy word to accumulator	5-65
HE	List available commands	5-28
HM	Hex mode	5-76

IBn	If accumulator is below	5-47
ICn	If accumulator bit is clear	5-48
IEn	If accumulator is equal	5-48
IFn	If channel is off	5-49
IGn	If accumulator is greater	5-49
ILn	Set integration limit	5-8
IMn	Integration mode	5-8
INn	If channel is on	5-50
IPn	Interrupt on absolute position	5-50
IRn	Interrupt on relative position	5-51
ISn	If accumulator bit is set	5-51
IUn	If accumulator is unequal	5-52
JPn	Jump to command absolute	5-39
JRn	Jump to command relative	5-39
LFn	Limit switch off	5-9
LMn	Limit switch mode	5-9
LNn	Limit switch on	5-10
LPn	Learn point	5-57
LTn	learn target	5-57
MAn	Move absolute	5-21
MCn	Macro call	5-40
MDn	Macro define	5-40
MF	Motor off	5-21
MJn	Macro jump	5-41
MN	Motor on	5-22
MRn	Move relative	5-22
NO	No operation	5-77
ODn	Output deadband	5-10
OOn	Output offset	5-11
PCn	Prompt character	5-77
PBn	Copy accumulator to byte	5-66
PEn	Set record period	5-29
PHn	Set servo output phase	5-11
PLn	Copy accumulator to long word	5-66
PM	Position mode	5-23
PRn	Position record enable	5-29
PWn	Copy accumulator to word	5-67
QM	Torque mode	5-23
RAn	Copy register to accumulator	5-67
RBn	Servo table byte to accumulator	5-68
RLn	Servo table long word to accumulator	5-68
RMn	Reset memory	5-41
RPn	Repeat command string	5-52
RT	Reset MPU	5-78
RWn	Servo table word to accumulator	5-69
SAn	Set acceleration	5-12
SDn	Set derivative gain	5-12
SEn	Set following error	5-13
SFn	Slave input format	5-13
SGn	Set proportional gain	5-14
SIn	Set integral gain	5-14
SLn	Accumulator shift left	5-69

SMn	Slave mode	5-24
SOn	Slave offset	5-15
SQn	Set torque	5-15
SRn	Accumulator shift right	5-70
ST	Stop velocity	5-24
SVn	Set velocity	5-16
TAn	Tell analog	5-30
TB	Tell breakpoint	5-30
TCn	Tell channel	5-31
TD	Tell derivative gain	5-31
TE	Tell command error	5-32
TF	Tell following error	5-32
TG	Tell proportional gain	5-33
TI	Tell integral gain	5-33
TL	Tell integration limit	5-34
TMn	Tell macro	5-42
TO	Tell optimal	5-34
TP	Tell position	5-35
TQ	Tell torque	5-35
TRn	Tell register	5-70
TS	Tell servo status	5-36
TT	Tell target position	5-36
TV	Tell command velocity	5-37
TX	Tell contour index	5-37
TZ	Tell servo index	5-38
UDn	Update display	5-71
VE	Tell firmware version	5-38
VGn	Velocity gain	5-16
VM	Velocity mode	5-25
WAn	Wait for time	5-53
WBn	Accumulator to servo table byte	5-71
WFn	Wait for channel off	5-53
WLn	Accumulator to servo table long word	5-72
WNn	Wait for channel on	5-54
WPn	Wait for absolute position	5-54
WRn	Wait for relative position	5-55
WSn	Wait for stop	5-55
WTn	Wait for target	5-56
WWn	Accumulator to servo table word	5-72
XFn	X-off code	5-78
XNn	X-on code	5-79

Parameter	Size	Related Commands	Table Offset	Default Setting	Comments
Servo Status	32 bits	TS	0 (0 hex)	(9000020A h)	Bit field, See note 1.
Encoder Position	32 bits	TP	4 (4 hex)	0	+/- Quadrature counts
Encoder Index Position	32 bits	TZ	8 (8 hex)	0	+/- Quadrature counts
Encoder Offset	32 bits	SO	12 (C hex)	0	+/- Quadrature counts
Breakpoint Position	32 bits	TB, IP, WP	16 (10 hex)	0	+/- Quadrature counts
Move Target	32 bits	TT, MA, MR	20 (14 hex)	0	+/- Quadrature counts
(reserved)	16 bits	*	24 (18 hex)	*	*
Trajectory Position	32 bits	TO	26 (1A hex)	0	+/- Quadrature counts
(reserved)	32 bits	*	30 (1E hex)	*	*
Trajectory Velocity	32 bits	TV	34 (22 hex)	0	16 +16 Bit fractional
Velocity Limit	32 bits	SV	38 (26 hex)	1000000	16 +16 Bit fractional
Acceleration Rate	32 bits	SA	42 (2A hex)	5000	16 +16 Bit fractional
Deceleration Rate	32 bits	DS	46 (2E hex)	5000	16 +16 Bit fractional
(reserved)	32 bits	*	50 (32 hex)	*	*
Aux. Encoder Position	32 bits	AT	54 (36 hex)	0	+/- Quad/Step counts
(reserved)	32 bits	*	58 (3A hex)	*	*
Slave Mode Ratio	32 bits	SM	62 (3E hex)	0	4 + 28 Bit fractional
Contour Point Position	12 bits	TX,CM	66 (42 hex)	0	Point number index
Proportional Gain	16 bits	SG, TG	70 (46 hex)	200	Integer value
Derivative Gain	16 bits	SD, TD	72 (48 hex)	200	Integer value
Integral Gain	16 bits	SI, TI	74 (4A hex)	5	Integer value
Velocity Feedforward	16 bits	VG	76 (4C hex)	0	Integer value
Accel. Feedforward	16 bits	AG	78 (4E hex)	0	Integer value
Decel. Feedforward	16 bits	DG	80 (50 hex)	0	Integer value
Following Error	16 bits	TF	82 (52 hex)	0	Integer value
(reserved)	16 bits	*	84 (54 hex)	*	*
(reserved)	16 bits	*	86 (56 hex)	*	*
(reserved)	16 bits	*	88 (58 hex)	*	*
(reserved)	16 bits	*	90 (5A hex)	*	*
(reserved)	16 bits	*	92 (5C hex)	*	*
Integration Limit	16 bits	IL, TL	94 (5E hex)	50	Integer value
Torque Limit	10 bits	SQ	96 (60 hex)	1023	Magnitude only
Max, Following Error	16 bits	SE	98 (62 hex)	1024	+ Quadrature counts
Output Offset	11 bits	OO	100 (64 hex)	0	Mag. + Direction
Target Deadband	16 bits	DB	102 (66 hex)	0	+ Quadrature counts
Torque Output	11 bits	TQ	104 (68 hex)	0	Mag. + Direction
Output Deadband	10 bits	OD	106 (6A hex)	0	Magnitude only
Delay at Target	16 bits	DT	108 (6C hex)	0	Integer value
Derivative Period	7 bits	FR	110 (6E hex)	0	Integer value
Direction Deadband	8 bits	DB	111 (6F hex)	0	Integer value
Breakpoint channel	8 bits	BC	112 (70 hex)	0	Integer value
Slave Format	8 bits	SF	113 (71 hex)	0	Bit field

Note 1. See TS command in Section 5 for full decoding. "9000020A" hex is "Motor off", "Position Mode" selected, and Limit inputs disabled. Although disabled, Limit inputs read as active if nothing is wired to them.

This manual applies to DSC firmware revision levels beginning at 1.0. The following table summarizes significant firmware modifications by release and date. The "VE" command can be entered at the DSC prompt to verify installed firmware revision level.

Revision	Date	Level	Notes
Rev 0.5	05/95	Beta unit release.	Initial firmware release. Firmware is specific to Beta level hardware.
Rev 1.0	08/96	Production release	1. Added input channel numbers to CH and CL commands. Created channel 10 to represent fault relay output. 2. Enabled DO and DQ recording functions. 3. Created GA parameter structure for accessing actual current and motor temperature. 4. Corrected AG and DG parameter polarity. 5. Modified multidrop code to disable all transmitters when communicating globally and to enable transmitter for point to point 422/485. 6. Modified Servo Status bit assignments, fault recovery and hardware reset logic. See appendix D
Rev 1.5	08/96	(Multitask Upgrade - Beta)	DSC2000 Multitasking Upgrade - Beta firmware available. Call Automotion for advance information.

(Sorry, this appendix is presently under development. 8/20/96)

1. If assistance is required call Automotion @ (313) 662-7771

(Sorry, this appendix is presently under development. A comprehensive list of DSC to SP motor switch setting will be available soon. 8/20/96)

1. If assistance is required call Automotion @ (313) 662-7771

The following tables may be used to understand how the feedback signals from the three rotor positional sensors are decoded and subsequently used to energize output motor phases. Commutation sensors may be optical or magnetic hall type devices.

In each of the tables below, a "1" indicates a high signal level at P2-10, P2-11 and, or P2-12. A "0" indicates a low logic level. The term "Source" refers to the condition whereby a particular motor phase is electrically connected via the DSC output bridge to the "positive" side of the DSC line rectified DC power supply. "Source" means that current is flowing from this supply connection into a given motor phase. Conversely, the term "Sink" refers to the condition whereby a given motor phase is electrically connected to the "negative" side of this supply, and current is flowing from a given motor phase to this connection

Supply "positive" and "negative" are relative to each other, and NOT relative to frame ground. This supply is NOT isolated from, and IS floating on the AC line. The terms "forward" and "reverse" are arbitrary and may yield either CW or CCW shaft motion depending on the motor and manufacture selected.

120 Electrical Degree Commutation State Table

Applies to model numbers: DSCxxxx-0xx-xxxx

Comm. Sensors			**Forward Motion**		**Reverse Motion**	
S1	S2	S3	(+) Source	(-) Sink	(+) Source	(-) Sink
0	0	1	MTR 2	MTR 1	MTR 1	MTR 2
1	0	1	MTR 2	MTR 3	MTR 3	MTR 2
1	0	0	MTR 1	MTR 3	MTR 3	MTR 1
1	1	0	MTR 1	MTR 2	MTR 2	MTR 1
0	1	0	MTR 3	MTR 2	MTR 2	MTR 3
0	1	1	MTR 3	MTR 1	MTR 1	MTR 3

Note: To change default rotation direction swap S1 and S3, swap MTR 2 and MTR 3 and swap encoder channels (A+!A) with (B+!B)

60 Electrical Degree Commutation State Table

Applies to model numbers: DSCxxxx-1xx-xxxx

Comm. Sensors			**Forward Motion**		**Reverse Motion**	
S1	S2	S3	(+) Source	(-) Sink	(+) Source	(-) Sink
0	0	0	MTR 2	MTR 1	MTR 1	MTR 2
1	0	0	MTR 2	MTR 3	MTR 3	MTR 2
1	1	0	MTR 1	MTR 3	MTR 3	MTR 1
1	1	1	MTR 1	MTR 2	MTR 2	MTR 1
0	1	1	MTR 3	MTR 2	MTR 2	MTR 3
0	0	1	MTR 3	MTR 1	MTR 1	MTR 3

Note: To change default rotation direction swap S1 and S3, swap MTR 1 and MTR 2 and swap encoder channels (A+!A) with (B+!B).

(Sorry, this appendix is presently under development. 8/20/96)

1. If assistance is required call Automotion @ (313) 662-7771

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP