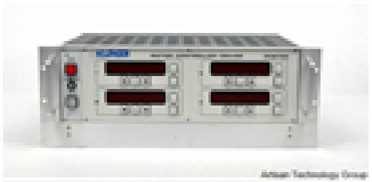

Artisan Technology Group

DCS750

DC MOTOR CONTROLLER

USER'S MANUAL

revision 2.07.05

KLINGER SCIENTIFIC CORPORATION

999 Stewart Avenue ■ Garden City, NY 11530 ■ Tel. No. 516-745-6800

Fax No. 516-745-6812

DCS750

DC MOTOR CONTROLLER

USER'S MANUAL

revision 2.07.05

The information in this document has been carefully checked and is believed to be entirely reliable. However, no responsibility is assumed for inaccuracies or inadvertent omissions. Furthermore, KLINGER SCIENTIFIC CORPORATION reserves the right to make changes to any product herein to improve reliability, function, or design. KLINGER SCIENTIFIC CORPORATION does not assume any liability arising out of the application or use of any product or circuit described herein.

TABLE OF CONTENTS

CHAPTER 1
GENERAL INFORMATION

1.1 INTRODUCTION

This manual provides general information, operating instructions, and functional description for the DCS750 DC Motor Controller.

The DCS750 is an integrated 4 axis programmable DC motor controller and amplifier system. Two popular interfaces, IEEE-488 and RS-232-C, are provided for host-to-DCS750 communications. Front panel alphanumeric displays and remote pushbutton motion switches are optional.

1.2 FEATURES

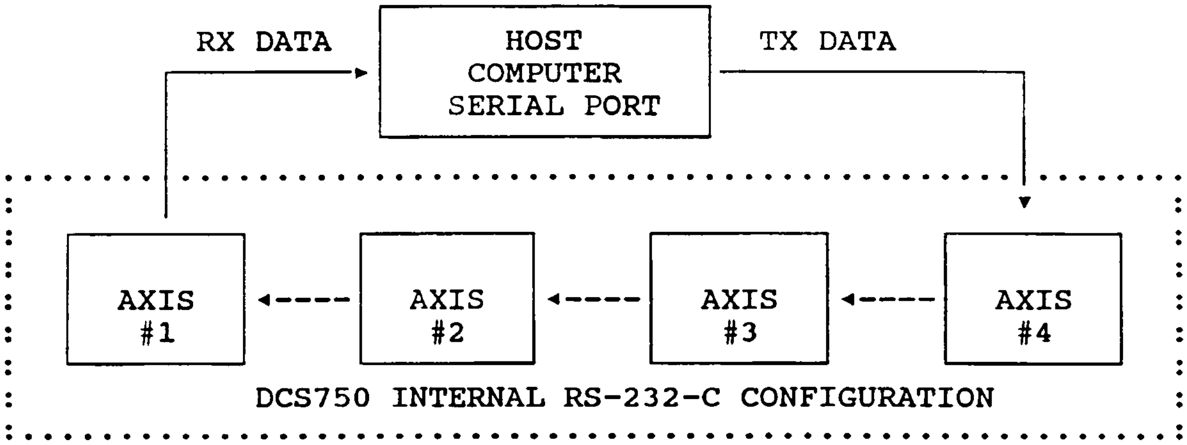
features include:

* IEEE-488 talk/listen interface compatibility
* RS-232-C full duplex interface (diasy-chained up to 30 axes)
* 24k non-volatile program storage (6k per axis)
* Internal program editor
* Real-time programmable host interrupts
* External event synchronization
* Absolute and relative positioning to ±10^9 counts
* Programmable motor power cutoff on excessive following error
* Local jog function capability
* TTL, differential and open collector encoder inputs
* Alphanumeric position/status displays
* Encoder quadrature fault detection
* Programmable travel limits to ±10^9 counts
* Velocity, position, and filter parameters may be changed on-the-fly
* Advanced instruction set with over 60 commands
* Trapezoidal velocity profile generator
* Automatic motor power cutoff on hardware travel limits
* Programmable digital PID filter with 16-bit coefficients
* 256uS sampling interval
* Precision homing capability
* Velocity and position modes of operation
* Velocity rates to 1,000,000 counts per second
* 2048 byte (512 per axis) command input buffer

1.3 GENERAL DESCRIPTION

The DCS750 provides precise control of up to four DC motors/positioners to achieve complex multi-axis positioning. Each axis consist of one controller module and amplifier.

Each axis of control has its own RS-232-C interface yet only one connector is provided for host-to-DCS750 hookup. This is accomplished by an internally *daisy-chained* serial link which permits up to four controller boards per system to operate over the same RS-232-C communication port. This scheme may be extended to include multiple DCS750 systems for a maximum of 30 axes.


RX DATA
HOST COMPUTER SERIAL PORT
TX DATA
AXIS #1
AXIS #2
AXIS #3
AXIS #4
DCS750 INTERNAL RS-232-C CONFIGURATION

The DCS750 implements two letter commands preceded by an axis call number (i.e., "3MO"). A call number is assigned to each axis via dip switches which reside on the controller module itself. The axis call number (also used for IEEE-488 device addressing) is factory set to 1, 2, 3, and 4 for axes 1 through 4, respectively.

The DCS750 is a standard rack-mount unit, 19" wide and 7" high. A Motherboard serves as the internal bus and wiring harness. The number of slide-in cards plugged into the Motherboard depends upon the number of positioners to be controlled.

System compensation is typically performed at the factory. The DCS750 should be properly configured and compensated for the positioner/motor combination in use (See section 6.5). The positioner motor assembly and DCS750 rear panel motor connector(s) should also have matching labels for easy identification and pairing. The suggested operating PID and trajectory parameters are stored in non-volatile memory and are automatically reloaded after a reset or power-cycle.

CHAPTER 2
SYSTEM HOOK-UP

2.1 DEVICE ADDRESSING

Each axis requires a unique IEEE-488 and RS-232-C device address, or call number. This can be any number from 1 to 30. The address setting is in binary form and can be selected via dip switch bank *S1* located on each controller board. Specifically, the ON position corresponds to a binary "1" and the MSB through LSB conform to S1-1,2,3,4 and 5, respectively. See Table 2-1.

NOTE: REMOVING THE SIX CAPTIVE SCREWS LOCATED ON THE FRONT PANEL WILL ACCESS THE CONTROLLER BOARD ADDRESS SWITCHES IF NECESSARY.

Like all standard devices connected to the IEEE-488 bus, the DCS750 will not respond until its device address is called. The device address selected also affects the RS-232-C communication link. Each axis is internally daisy-chained to its neighboring axis and requires a call number.

NOTE: EACH UNIT IS FACTORY SET TO RESPOND TO ADDRESS 1, 2, 3, AND 4 FOR AXIS 1, 2, 3, AND 4.

WARNING: BEWARE OF AXES WITH DUPLICATE DEVICE ADDRESSES !!!

AXIS ADDRESS	S1-1	S1-2	S1-3	S1-4	S1-5	AXIS ADDRESS	S1-1	S1-2	S1-3	S1-4	S1-5
1	OFF	OFF	OFF	OFF	ON	16	ON	OFF	OFF	OFF	OFF
2	OFF	OFF	OFF	ON	OFF	17	ON	OFF	OFF	OFF	ON
3	OFF	OFF	OFF	ON	ON	18	ON	OFF	OFF	ON	OFF
4	OFF	OFF	ON	OFF	OFF	19	ON	OFF	OFF	ON	ON
5	OFF	OFF	ON	OFF	ON	20	ON	OFF	ON	OFF	OFF
6	OFF	OFF	ON	ON	OFF	21	ON	OFF	ON	OFF	ON
7	OFF	OFF	ON	ON	ON	22	ON	OFF	ON	ON	OFF
8	OFF	ON	OFF	OFF	OFF	23	ON	OFF	ON	ON	ON
9	OFF	ON	OFF	OFF	ON	24	ON	ON	OFF	OFF	OFF
10	OFF	ON	OFF	ON	OFF	25	ON	ON	OFF	OFF	ON
11	OFF	ON	OFF	ON	ON	26	ON	ON	OFF	ON	OFF
12	OFF	ON	ON	OFF	OFF	27	ON	ON	OFF	ON	ON
13	OFF	ON	ON	OFF	ON	28	ON	ON	ON	OFF	OFF
14	OFF	ON	ON	ON	OFF	29	ON	ON	ON	OFF	ON
15	OFF	ON	ON	ON	ON	30	ON	ON	ON	ON	OFF

Table 2-1

2.2 TOP ZERO DISABLE

The Top Zero (index) signal is used in conjunction with the origin switch to guarantee the repeatability of the home position. This signal is generated from electro-optical components, and a disk, mounted on the motor shaft. The Top Zero signal is calibrated to become true well after the origin switch closes, thereby eliminating switch trip-point inaccuracies. However, not all positioners equipped with the switch origin search option require or provide a Top Zero signal. Thus, you may set dip switch S1-6 ON to disable or OFF to enable this feature. For more information on origin search see section 3.3.

NOTE: MOST POSITIONERS EQUIPPED WITH AN ORIGIN SWITCH ALSO PROVIDE A TOP ZERO.

2.3 DCS750 TO POSITIONER HOOK-UP

Now, we are ready to hook-up the DCS750 to the positioner(s). To interconnect the positioner motor to the amplifier module a DCS750-to-motor cable assembly is required. The side of the cable assembly with the round, 19 pin, female connector, mates with the positioner. The opposite end with the D shaped, 25 pin, male connector mates with the DCS750.

CAUTION: DO NOT MISTAKE THE RS-232-C CONNECTOR FOR THE MOTOR CONNECTOR !!!

2.4 LINE VOLTAGE INPUT

The line voltage necessary to properly operate this system is indicated on the rear panel identification label. If your system is not properly configured for the desired operating line voltage please contact the factory for conversion procedures.

2.5 HOST TO DCS750 HOOK-UP

With the front panel POWER switch in the OFF position, insert the line cord supplied into the line cord receptacle.

Host to DCS750 interfacing is accomplished via a standard IEEE-488 or RS-232-C cable (See appendix A.1 and A.2). Carefully insert the mating cable plug into the labeled DCS750 rear panel IEEE-488 or RS-232-C socket. Connect the opposite end of the cable to the host computer. This completes the installation process for the DCS750.

2.6 RETROFITTING CONTROLLER MODULES

The DCS750 is a modular, slide-board-in type, system making it possible to perform field upgrades. As a case study, we will now examine the steps necessary to upgrade the DCS750 system from two(2) axes to three(3) axes. Briefly, it will be necessary to install an additional 60 Volt power supply(#87.09.04.103), controller(#88.09.09.11), and amplifier module(#87.09.04.158 or #88.09.04.194). Note that one(1) 60 Volt power supply module typically serves two(2) axes.

UPGRADING A 3rd AXIS

STEP	PROCEDURE
1	Turn the system power off!
2	Carefully, remove the front panel using the six(6) captive screws.
3	If your system is equipped with the alphanumeric position display option it will be necessary to disconnect each forty(40) pin ribbon connector from the front panel display board.
4	Remove the vertical phenolic bracket held by two(2) screws and located near the center.
5	Look carefully inside the enclosure to the rear mother board. Find the blue RS-232-C daisy-chain jumpers. These jumpers permit the RS-232-C signal to bypass non-existing axes. And since we are adding a new axis at least one(1) jumper position will be moved. The blue jumper near the center corresponds to axis #3. Carefully shift the jumper from the first two(2) pins on the left side over to the last two(2) pins on the right. Now, visually verify the connection before proceeding.
6	Insert the 60 Volt power supply #87.09.04.103 into the center most slot. (assuming a 2nd power supply module does not already exist there).
7	Insert the controller module #88.09.09.11 into the slot just to the right of the 2nd power supply. If you are upgrading the display option also make sure that the forty(40) pin controller-to-display board ribbon is already inserted into J4 of the controller module.
8	Carefully, insert the amplifier #88.09.04.194 (200 watt) or #87.09.04.158 (20 watt) into the slot just to the right of the 3rd controller (of step 7).
9	Make certain that all of the modules installed in steps 5 through 8 are properly fastened. Push the plastic handle on each module in and down to secure the boards.
10	If you are not upgrading the display option skip over to step 18.

STEP PROCEDURE

11 To upgrade the display board from two(2) axes to three(3) axes first remove the corresponding blue jumper located on the display board itself.

12 Insert the forty(40) pin ribbon connectors already mated to the controller into its respective socket on the display board. Use the pin 1 to pin1 convention indicated on the sockets and connectors. Note that the ribbon cable should not be twisted--straight connection from board to board only.

13 Carefully re-attach the front panel assembly onto the DCS750 enclosure and fasten with the six(6) captive screws.

That completes the DCS750 three(3) axes field upgrade. The overall procedure for a two(2) or four(4) axis upgrade is the same. However, if you need further assistance please contact the factory.

CHAPTER 3
LOCAL FRONT PANEL OPERATION

3.1 LED STATUS INDICATORS

Four green LEDs located on each controller board provide basic local status information useful during troubleshooting or configuration. Carefully unscrewing the six front panel captive screws will gain access to the boards and LEDs should it become necessary.

> IMPORTANT: USERS WITH DCS750 SYSTEMS EQUIPPED WITH THE FRONT PANEL POSITION DISPLAY OPTION MAY PROCEED TO SECTION 3.2

LABEL	LED	STATUS INFORMATION
ACTIVE	ON	Controller module powered.
	OFF	Controller module not powered.
READY	ON	Origin search active
	OFF	Origin search idle
+LIMIT	ON	+ direction travel *hardware* limit active.
	OFF	+ direction travel *hardware* limit not active
-LIMIT	ON	- direction travel *hardware* limit active.
	OFF	- direction travel *hardware* limit not active.

Table 3.1

If the +LIMIT LED is lit the controller is NOT allowed to perform any motion commands in the positive direction. If the -LIMIT LED is lit the controller is NOT allowed to perform any motion commands in the negative direction.

NOTE: 1) BOTH LIMIT LEDS WILL BE LIT IF THE MOTOR IS NOT CONNECTED.
2) YOU MAY SKIP SECTION 3.2, 3.3, AND 3.4 IF YOUR SYSTEM IS NOT EQUIPPED WITH THE FRONT PANEL POSITION DISPLAY OPTION.

3.2 FRONT PANEL POSITIONING

Front panel control of movement and positioning is done by the following:

- right arrow switch ("--►") does low speed (jog) motion in the positive direction.
- left arrow switch ("◄--") does low speed (jog) motion in the negative direction.
- **HIGH SPEED** switch does high speed motion when pressed along with a low speed switch.

Manual positioning consist of the following steps:

1. To move a "single step" at a time on the axis selected, press and release one of the arrow switches.

   NOTE: The "single step" is programmable with the "SS" command (see chapter 6). The default is <u>*one*</u> *encoder increment or decrement.*

2. To move continuously at low speed on the axis selected, press and hold down one of the arrow switches.

3. To move continuously at high speed, press and hold down both an arrow switch and the **HIGH SPEED** switch.

3.3 ORIGIN SEARCH

The Origin Search function is an automatic home seeking operation. It returns a positioner/motor to a preset position-a reference point for subsequent positioning. The origin point must be defined and conveyed to the DCS750 with the "OR" (origin search) command (see chapter 6) and dip switch S1-6 (Top Zero).

The three possible origin points are:

1. The position defined by the location of a switch mounted on a positioning stage.

2. The position defined by the location of a switch mounted on a positioning stage in conjunction with a top zero (index) signal.

3. The position that corresponds to the step count of zero on the display for that axis.

Origin point number 3 is called a floating point origin because a new position can always be defined as the origin point by simply moving to the new location and resetting the position to zero. Note that local origin search activation can be disabled by setting "DO" command bit-1.

3.4 POSITION RESET

Each axis has its own position reset switch. To perform a reset, simply press and release the RESET switch located to the right of the display you wish to reset. This switch is functionally equivalent to the "DH" command. Note that local reset activation can be disabled by setting "DO" command bit-2.

Resetting a particular axis display does three things:

1. Resets the position display readout to zero

2. Sets the present position as the floating point origin

3. Sets the present position as the reference point for the absolute positioning and software limit commands (see chapter 6)

IMPORTANT: IF THE POSITION RESET BUTTON IS DEPRESSED WHILE A MOTION AND/OR PROGRAM IS BEING EXECUTED A MOTION/PROGRAM BREAK IS GENERATED AND THE MESSAGE "STOP" WILL BE MOMENTARILY DISPLAYED.

3.5 LOW AND HIGH SPEED PROGRAMMING

Programming the low and high speeds for a particular axis involves a two switch combination. One of the switches used is the position RESET switch (see section 3.3). The other switch is either arrow button for low speed adjustment or the HIGH SPEED button for high speed alteration. These speeds, once set, are retained in non-volatile memory and are automatically reloaded after reset or power-cycle. Note that local speed programming can be disabled by setting "FO" command bit-2.

IMPORTANT: THE JOG AND ORIGIN SEARCH LOW AND HIGH SPEEDS ARE ALSO SET WITH THE "OL" AND "OH" COMMANDS. HOWEVER, ONLY THE "OA" COMMAND CONTROLS JOG AND ORIGIN SEARCH ACCELERATION.

SETTING THE LOW SPEED RATE

To set the low speed for any axis, do the following:

1. Press and hold down the position RESET switch
2. Press and hold down either arrow switch
3. Release the position RESET switch
4. Release the arrow switch

At this time, the display will show the present low speed setting. The display will be flashing to indicate that you are in the speed set mode.

5. To increment the low speed by 100 counts/second, press and release the right arrow switch. For continuous incrementing, hold down the right arrow switch.
6. To decrement the low speed by 100 counts/second, perform step 5 with the left arrow switch.
7. To increment/decrement the low speed by 2000 counts/second use the HIGH SPEED switch. First you must establish increment or decrement by pressing the appropriate arrow switch once. Then press the HIGH SPEED switch to change by 2000 counts/second.
8. When you have reached your desired speed, press and release the position RESET switch to save your low speed setting and exit speed set mode.

NOTE: PRESS AND RELEASE THE POSITION RESET SWITCH TO EXIT SPEED SET MODE

SETTING THE HIGH SPEED RATE

To set the high speed for any axis, do the following:

1. Press and hold down the position RESET switch
2. Press and hold down the HIGH SPEED switch
3. Release the position RESET switch
4. Release the HIGH SPEED switch

> At this time, the display will show the present high speed setting. The display will be flashing to indicate that you are in the speed set mode.

5. To increment the high speed by 100 counts/second, press and release the right arrow switch. For continuous incrementing, hold down the right arrow switch.
6. To decrement the high speed by 100 counts/second, perform step 5 with the left arrow switch.
7. To increment/decrement the high speed by 2000 counts/second use the HIGH SPEED switch. First you must establish increment or decrement by pressing the appropriate arrow switch once. Then press the HIGH SPEED switch to change by 2000 counts/second. When you have reached your desired speed, press and release the position RESET switch to save your high speed setting and exit speed set mode.

NOTE: PRESS AND RELEASE THE POSITION RESET SWITCH TO EXIT SPEED SET MODE

3.6 SYSTEM RESET

The system RESET switch, not to be confused with the position RESET switch located to the right of each position display, is directly coupled to all internal power-up reset circuits. This switch is located above the POWER button at the upper left corner. Depressing and releasing it causes a system "cold-start" which disables all motor power, re-initializes internal peripherals, clears position displays, and reloads trajectory parameters.

IMPORTANT: THIS SWITCH AFFECTS ALL AXIS INSTALLED WITHIN THE SYSTEM.
(So use it wisely!)

3.7 PROGRAM SELECTION AND EXECUTION

The DCS750 uses the front panel "HIGH SPEED" switch (which is normally used for high speed jogging) to invoke local program selection mode. In this mode, the DCS750 permits up to 99 programs to be accessed for execution provided that they are compiled (see "CP" command in Chapter 6). Note that local program activation can be disabled by setting "FO" command bit-2.

SELECTING THE PROGRAM

To select a program for any axis, do the following:

1. Press and hold down the HIGH SPEED switch

> After approximately 2 seconds the display will prompt you with "PROGRM 0" to indicate that you are in program selection mode.

2. Release the HIGH SPEED switch
3. To increment the program selector, press and release the right arrow switch. For continuous incrementing, hold down the switch.
4. To decrement the program selector, press and release the left arrow switch. For continuous decrementing, hold down the switch.

EXECUTING THE PROGRAM

To execute any selected program simply press and release the HIGH SPEED switch again.

> NOTE: THE USER MAY AT ANYTIME EXIT THIS MODE BY SIMPLY PRESSING AND RELEASING THE POSITION RESET SWITCH.

3.8 DISPLAY MESSAGES

MESSAGE DESCRIPTION

VER n,n After a system reset or power-cycle each axis reports the revision level of the firmware in use. i.e., "VER n,n" for revision n.n

AXIS nn After a system reset or power-cycle each axis reports its current axis address as configured by dip switch bank S1.
NOTE THAT EACH AXIS MUST HAVE A UNIQUE DEVICE ADDRESS

ENCODER Built-in circuitry which monitors encoder signal timing characteristics for each axis displays the message "ENCODER" when a fault is detected. Typically, this is due to either excessive signal noise or poor quadrature symmetry. *NOTE THAT "FO" COMMAND, BIT-1, DISABLES THIS FEATURE.*

TOO FAST When the positioner/motor is not able to *FOLLOW* or keep-up with the programmed target trajectory the message "TOO FAST" is displayed. The "FE" command sets the trip-point or number of encoder counts the axis is allowed to "fall behind" before the motor is turned off and the message is displayed. There are three(3) typical causes for this error condition:

1) The motor or encoder phasing may be inverted or non-standard. This may be corrected with the "IP" command or interchanging motor +/- leads or interchanging encoder channel A and B.
2) The target velocity and/or acceleration may not be suitable or realistic for the mechanical system.
3) The motor amplifier and/or power supply may not be operable. Check to see that they are properly fastened into the backplane, or have blown fuses.

If the problem persist consult with the factory.

LIMIT This message will appear whenever the axis has encountered a software or hardware limit in either direction. In controller module revision C and later front panel jog pushbuttons will remain lit whenever a hardware limit is encountered. For example, the "->" (right arrow) positive direction jog pushbutton will light-up when the positive hardware travel limit is encountered.

NO MOTOR If both hardware travel limit inputs (motor connector pins 17 & 18) are not grounded the axis assumes no motor exist and does not perform any motion commands. In controller module revision C and later, both front panel jog pushbuttons (right and left arrows) remain lit as long as no motor is connected to that axis.

MESSAGE DESCRIPTION

STOP A motion/program BREAK occurs and the message "STOP" appears whenever the axis "reset" pushbutton is depressed and released while a motion is being performed or a program is being executed. However, the motor remains in the *ON* state.

E STOP A motion/program BREAK occurs and the message "E STOP" appears whenever the axis Emergency Stop input is triggered via the rear panel GPIO connector. *NOTE THAT THE MOTOR IS TURNED-OFF*

ZERO Depressing and releasing the axis "reset" pushbutton while that axis is idle causes the position register to be set to absolute position zero and display the message "ZERO". This is functionally equivalent to the "DH" command.

EXIT Depressing and releasing the axis "reset" pushbutton while in front panel speed setting mode or front panel program selection mode causes the axis to save any changed parameters and *exit* that environment.

HI SPEED Depressing and releasing the axis "reset" and "high speed" pushbuttons as described in section 3.5 causes the axis the enter the front panel jog high speed programming environment.

LO SPEED Depressing and releasing the axis "reset" and either right or left arrow pushbuttons as described in section 3.5 causes the axis the enter the front panel jog low speed programming environment.

PROGRM n Depressing the axis "high speed" pushbutton for 2 to 3 seconds as described in section 3.7 causes that axis to enter front panel program selection and execution mode.

PRG MODE Whenever an axis receives the "EP" (Enter Programming mode) command this message appears and remains until a percent ("%") character-carriage return combination (end of programming mode) is received.

DISP OFF If bit-6 of the "FO" (Format Output) command is set (i.e.,"FO64") position display updating is disabled and this message will appear. Reset bit-6 to restore position display updating. This message also momentarily appears during "TV" (Tell Velocity) command execution.

HEX MODE This message indicates that the axis is in the HeXidecimal interface mode. See section 6.7

PURGE During system reset each axis monitors the front panel right and left arrow jog switches. If both pushbuttons are depressed simultaneously during system reset the axis will display the message "PURGE". Once the pushbuttons are released internal memory is purged and default motion parameters are installed. This message momentarily appears during "XX" (Purge memory) command execution.

MESSAGE DESCRIPTION

BAD U2 This message may momentarily appear during "XX" (Purge memory) command execution or automatic memory purging. Possible hardware failure at integrated circuit U2 (RAM) may exist if this message persist.

BAD U15 Possible hardware failure may exist at integrated circuit U15 (servo controller) if this message persist.

DISABLED This message indicates that the desired local origin search, jog, or position reset function is currently disabled with the "DO" command.

CHAPTER 4
RS-232-C INTERFACING

4.1 INTRODUCTION

The DCS750 supports up to four axes of DC motors/positioners to achieve complex multi-axis positioning. Each axis consists of one controller with a unique address (factory set to 1, 2, 3, & 4). Each axis is also internally daisy-chained to its neighboring axis via factory installed jumpers located on the motherboard (see section 2.6). This permits communication to all axes via a single RS-232-C port (see figure on page 1-2).

The DCS750 supports a full duplex (able to transmit and receive simultaneously) RS-232-C interface. A simple terminal emulation program provides the easiest tool for evaluating the system.

The system accepts two letter commands preceded by an axis number (i.e., "3PA+100") and terminated with a carriage return (ASCII 13) or carriage return/line feed (ASCII 10) combination.

When a new command is received without an axis number the instruction defaults to the last axis addressed. Therefore, the axis number is only necessary for the first command to a different axis.

Most commands, including target velocity and position updating, are accepted and executed *on-the-fly*. Also, all strings transmitted from the DCS750 to the host computer contain an axis identifying header. The header consists of the axis call number and an arrowhead ">" (ASCII 62).

i.e., **"04> +100 COUNTS"** (in response to, **"4TP"**)

The header always identifies the responding axis and can not be suppressed. The leading zero, spaces, and units (i.e., "COUNTS"), however, may be suppressed with the "FO" instruction to reduce command response time.

The DCS750 terminates each output string with a carriage return (ASCII 13)/line feed (ASCII 10) combination.

4.2 CONFIGURATION

The DCS750 uses a standard 25 pin female D connector for interfacing to the host computer. If the host computer is configured as a DTE like most PCs then a straight-through cable is all that is needed to get started (See appendix A.2).

CAUTION: DO NOT MISTAKE THE MOTOR CONNECTOR FOR THE RS-232-C CONNECTOR!!!

There are only three (3) signals used for the RS-232-C interface. Pin 2 receives data from the host, pin 3 transmits data to the host, and pin 7 is signal ground.

The RS-232-C interface is always configured for 8 data bits, no parity, and 1 stop bit. The baud rate is factory set to 9600 baud. However, after communication has been established the baud rate may be changed with the "SR" (set rate) command (see chapter 6). i.e., "SR 6" (4800 baud). Any changes in baud rate will not take affect until after the next system reset or power cycle.

CAUTION: ALL AXES MUST BE SET TO THE SAME BAUD RATE!!!

PARAMETER		BAUD RATE
0	<------>	110
1	<------>	150
2	<------>	300
3	<------>	600
4	<------>	1200
5	<------>	2400
6	<------>	4800
7	<------>	9600

The DCS750 serial interface is always in echo mode. Characters received are automaticaly transmitted back to the host. If the system is not echoing characters then double-check all interconnections from the host to the DCS750. If this problem persist please read section 2.6 for the proper setting of the internal RS-232-C "daisy-chaining" jumpers.

4.3 SOFTWARE HANDSHAKE

The RS-232-C interface doesn't use any handshake lines for its operation, so what ever device you are using for communication must be configured for operation without hardware handshaking.

The DCS750 implements *software* handshaking over the RS-232-C interface. An XOFF (ASCII 19) character is only transmitted whenever the 512 byte per axis input buffer is *near full*. In this case, after the command is processed an XON (ASCII 17) character is transmitted to the host.

It is worth noting that the DCS750 does not normally transmit an XOFF character upon receiving a command (i.e., "1PA-10000") and an XON after the command has been processed. Instead, so long as there's space in the buffer, commands are *stacked* in the order received and executed as soon as possible.

If your application involves down-loading relatively large blocks of program lines or immediate commands be sure to implement the following within your program:

STEP	DESCRIPTION
1	Begin sending commands until an **XOFF** (ASCII 19) is transmitted
2	Wait to receive an **XON** character (ASCII 17)
3	repeat step 1

NOTE THAT IN MOST APPLICATIONS THE BUFFER NEVER REACHES "NEAR FULL" STATUS.

Conversely, when the DCS750 is sending data (i.e., a program listing) it is checking for XON/XOFF characters from the host computer. If the host computer can not accept more data, an XOFF from the host to the DCS750 will suspend transmission until an XON is received.

4.4 TESTING THE RS-232-C INTERFACE

The single most useful concept to remember while programming, testing, and/or troubleshooting a DCS750 unit is that it behaves more like four single axis controllers enclosed together than it does a single four axes controller. Perhaps this concept may be somewhat confusing at this point, but as you interact with the system more and begin to create the application program it will become clearer.

When the RS-232-C cable is connected and power is applied you may begin to transmit commands to the DCS750 (see chapter 6). Type the "1VE" (tell VErsion) command terminated with a carriage return (ASCII 13) and send it to the DCS750. A firmware version response will verify correct operation of the communication link even if the system is not completely operational yet.

The DCS750 is factory set to generate a line feed to enhance display readability whenever a carriage return is detected. This feature can be disabled with the "FO" command at anytime.

Once again, all characters received are automatically echoed back to the host computer. Strings transmitted from the DCS750 to the host always lead with an identification header and terminate with a carriage return (ASCII 13)/line feed (ASCII 10) combination.

Remember that the DCS750 supports a full duplex (able to transmit and receive simultaneously) RS-232-C communication link. And a simple terminal emulation program provides the easiest tool for evaluating the system at this point.

If you do not have access to a simple RS-232-C terminal emulation program perhaps you may wish to use the sample programs written for PC compatibles in section 7.3 to test the communication link.

After the communication link has been established you may wish to proceed to Chapter 6 and familiarize yourself with the DCS750 command set.

CHAPTER 5
IEEE-488 INTERFACING

5.1 INTRODUCTION

We will now characterize the procedures required to establish communication between the DCS750 and your host computer via IEEE-488 (GPIB). The DCS750 is equipped with a GPIB talk/listen interface capable of receiving commands, reporting status, and generating Service ReQuest (SRQ) interrupts on the occurrence of an event.

5.2 LISTENER

To the GPIB controller (your host computer) the DCS750 system appears to be four unique GPIB devices. In fact, a DCS750 unit consist of up to four independent devices (or axes) housed together where each has its own unique address (or call number) and GPIB interface.

Each DCS750 unit provides a 2048 byte (512 per axis) command input buffer to relieve communication bottlenecks typical in multi-axis applications. This permits over-all faster data transfers between the GPIB controller and its devices. In addition, the DCS750 accepts and executes new commands on any axis from either RS232 or GPIB interface, even while running a program.

NOTE: AXIS CALL NUMBERS ARE NOT REQUIRED WHEN COMMUNICATING OVER THE GPIB

The axis call number is implicit with the GPIB interface. Their call numbers and GPIB device addresses are always equal because they are derived from the same on-board switch bank. Therefore, any application program developed must be able change the GPIB device address to gain access to multiple axes.

WHAT ABOUT EOS/EOI TERMINATIONS

Normally, DCS750 commands are terminated by a carriage return (ASCII 13). However, via the GPIB interface the firmware will also recognize a carriage return/line feed, carriage return/EOI, line feed/EOI, carriage return/line feed/EOI, or simply an EOI signal as a command terminator (*automatically*).

Also, commands may be entered in either upper or lower case characters, with or without spaces anywhere in the command line.

5.3 TALKER

The GPIB communication link is a half-duplex interface (data can not be transmitted and received simultaneously). To obtain the position status of axis #1, for example, first place the device <u>listen</u> address #1 on the bus. Then output the ASCII characters "TP" with a carriage return and/or EOI terminator. The DCS750 will now wait up to 10 seconds (before timing-out) for the host computer to place the device #1 talk address on the bus. The DCS750 as configured from the factory should output the following:

```
"01> +0 COUNTS"
```

Each string transmitted by the DCS750 to the host computer is terminated with a carriage return/line feed/EOI combination. However, several DCS750 commands need to output more than one line of information. The list program lines ("LL") command, for example, may transmit up to 9999 strings (provided a program that size existed). Normally, this is not a problem via the RS-232-C port, but the GPIB is only a half-duplex interface and more needs to be done.

First, the application program must distinguish between those commands that will be reporting information from those commands that will not so that we do not inadvertently assert the GPIB talk address and cause a device time-out. Next, the application program should distinguish between single and multi-line response commands. For multi-line response commands the application program will have to assert the GPIB talk address, take in the data, and repeat this process until all lines have been received. And some multi-line reporting commands have a fixed number of lines (i.e., "VE" command), while others do not (i.e., "LP" command).

To circumvent this dilemma each axis may be configured to transmit an "END" line or a multi-string terminator.

```
01> +0 COUNTS
01> END
```

The last line of any response will contain the sub-string "END" and can be easily tested for. Setting bit-4 of the "FO" command register enables this feature. For example, "FO 16" enables the multi-string terminator and "FO 0" disables it. (See "FO" command in Chapter 6).

IMPORTANT: **Send the question mark "?" (ASCII 63) command and assert the talk address to read the contents of the error message buffer.**

5.4 INTERRUPT GENERATION

Many devices on the IEEE-488 occasionally get themselves into trouble or need to inform the host of some important occurrence. If a positioner inadvertently runs into a limit (software or hardware), how does the computer know about it? Obviously, the host always has the option of polling the axis status with the "QS" and/or "MS" commands. However, polling up to four(4) axes may "tie-up" the host from performing other, perhaps important, tasks. In general, it is more efficient to have the device or, in this case, axis request for service as needed.

The IEEE-488 (GPIB) bus has 24 conductors or lines - eight data and sixteen other lines reserved for special use. One of these special lines is called the SRQ (Service ReQuest) line. Any device that needs a special job done can use this line to tell the host.

Since every device uses the same SRQ line, how does the host know "who-dun-it"? One way for the controller to determine this is to ask every device on the bus, one at a time. This is known as a serial poll. Every device that the host polls returns a status byte that tells the host: a) if that device or axis requested service and, if so, b) what kind of service that device requires.

The user may select the interrupt condition which will assert the SRQ line with the "SO" command. (see "SO" command in Chapter 6)

OPTION	SRQ CONDITION GENERATOR
0	DISABLE SRQ GENERATOR
1	MOTOR ERROR OR COMPLETED ORIGIN SEARCH
2	MOTOR ERROR OR COMPLETED PROGRAM
3	MOTOR ERROR OR COMPLETED MOTION
4	COMPLETED TIMER
5	HIGH-TO-LOW TRANSITION INPUT BIT (see Chapter 8)
6	LOW-TO-HIGH TRANSITION INPUT BIT (see Chapter 8)
7	MOTOR ERROR (see "MS" command in Chapter 6)
8	"ON" COMMAND TRIP POINT REACHED (see "ON" command)

A GPIB serial poll (or system RESET) will release the SRQ line. However, another occurrence of the same condition will again cause the SRQ line to be asserted. Only option zero (0) will disable continued SRQ generation. The SRQ option is *automatically* reloaded after RESET and power-cycle.

5.5 STATUS BYTE

In general, when an interrupt condition occurs and the SRQ line is asserted, the GPIB controller (the host computer) performs a serial poll on all devices that could have possibly caused the interrupt. The GPIB controller will receive a status byte from each device and test bit 6 (rsv). The device, or axis, which sets this bit is presumed to be the axis requesting service. The remaining bits each have a special meaning.

The DCS750 has four status bytes with different bit interpretations. SRQs generated with "SO" command options 1-6 use status byte #1 and option 7 uses status byte #2. The "RQ" (ReQuest for service) command generates a user programmable status byte #3. And option 8 always returns status byte #4 with a value of 255. (See "SO", "ON" and "RQ" commands in Chapter 6)

STATUS BYTE #1

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ORIGIN ACTIVE	RSV	VELOCITY MODE	INPUT BIT	TIMER ACTIVE	PROGRAM ACTIVE	MOTOR ERROR	MOTION COMPLETE

BIT	STATE	DEFINITION	
0	1	MOTION COMPLETE	
	0	" NOT COMPLETE	
1	1	MOTOR ERROR DETECTED (Status byte #2: bits 1-5 & bit 7 ORed)	
	0	" " NOT DETECTED	
2	1	INTERNAL PROGRAM ACTIVE	(see "EP","CP" & "EX" commands)
	0	" " NOT ACTIVE	
3	1	TIMER ACTIVE	(see "WT","WA" & "WS" commands)
	0	" NOT ACTIVE	
4	1	GPIO INPUT BIT LOW	(see "QS", "TS" and "SO" commands)
	0	" " " HIGH	
5	1	VELOCITY MODE ACTIVE	(see "MV" command)
	0	POSITION " "	(see "PR" & "PA" commands)
6	1	SERVICE REQUEST GENERATED	
	0	" " NOT GENERATED	
7	1	ORIGIN SEARCH ACTIVE	(see "OR" command)
	0	" " NOT ACTIVE	

STATUS BYTE #2

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
QUAD ERROR	RSV	FOLLOWING ERROR	+SOFTWARE LIMIT	-SOFTWARE LIMIT	+HARDWARE LIMIT	-HARDWARE LIMIT	MOTOR OFF

BIT	STATE	DEFINITION
0	1	MOTOR IS OFF (see "MF" command)
	0	MOTOR IS ON (see "MO" command)
1	1	NEGATIVE HARDWARE TRAVEL LIMIT ACTIVE
	0	" " " " IDLE
2	1	POSITIVE HARDWARE TRAVEL LIMIT ACTIVE
	0	" " " " IDLE
3	1	NEGATIVE SOFTWARE TRAVEL LIMIT ACTIVE (see "SL" & "TL" commands)
	0	" " " " IDLE
4	1	POSITIVE SOFTWARE TRAVEL LIMIT ACTIVE (see "SL" & "TL" commands)
	0	" " " " IDLE
5	1	EXCESSIVE MOTOR FOLLOWING ERROR (see "FE" & "TE" commands)
	0	FOLLOWING ERROR WITHIN RANGE
6	1	SERVICE REQUEST GENERATED
	0	" " NOT GENERATED
7	1	ENCODER QUADRATURE ERROR DETECTED (<200KHz operation only)
	0	NO ENCODER " " "

An occurrence of either a hardware or software limit, excessive motor following error, or encoder quadrature error is considered a *motor error* and will assert the SRQ line (if option 7 is selected) and set bit 1 of status byte #1.

NOTE: *NEW "PA" AND "PR" COMMANDS CLEAR ENCODER QUADRATURE AND EXCESSIVE FOLLOWING ERROR BITS.*

If bit 6 (rsv) is not set in the status byte after a serial poll the information retrieved is not reliable. Do not use the GPIB serial poll to obtain status information without generating SRQs. There are other reporting commands better suited for that purpose. (see chapter 6)

CHAPTER 6
PROGRAMMING

6.1 INTRODUCTION

The DCS750 implements two letter ASCII commands preceded by an axis number (i.e., "3MO"). The number-to-axis association is configured with on-board dip switches on each controller module. This axis number (also used for GPIB device addressing) is factory set to 1, 2, 3, and 4 for axes 1 through 4, respectively.

6.2 INSTRUCTION SET

MOTION

ASCII CODE	EXAMPLE	INTERPRETATION AND RANGE
AB	"AB"	Abort motion
AC	"AC 20000"	ACceleration (250-10^9 counts/sec^2)
MF	"MF"	Motor oFf
MO	"MO"	Motor On
MV	"MV+"	MoVe (velocity mode)
OA	"OA 20000"	Origin and jog Acceleration (250-10^9 counts/sec.^{2})
OH	"OH 11000"	Origin and jog High speed (1,000,000 counts/sec.)
OL	"OL 900"	Origin and jog Low speed (1,000,000 counts/sec.)
OR	"OR 1"	Origin Search (0=floating, 1 & 2=switch)
ST	"ST"	STop motion

MOTION (continued)

ASCII		
CODE	EXAMPLE	INTERPRETATION AND RANGE
PA	"PA -10000"	Position Absolute ($\pm 10^9$ counts)
PR	"PR +40000"	Position Relative ($\pm 10^9$ counts)
VA	"VA 5000"	Velocity Absolute (1,000,000 counts/sec)
VR	"VR -200"	Velocity Relative (±1,000,000 counts/sec)
VS	"VS 2237"	Sample time Velocity (16,777,778)

MOTION RELATED

ASCII		
CODE	EXAMPLE	INTERPRETATION AND RANGE
DH	"DH"	Define Home
DS	"DS 2"	Derivative Sampling interval (0-255)
FE	"FE 500"	Following Error (0-32000 counts)
IP	"IP 1"	Invert Phase (0=standard, 1=non-standard)
IL	"IL 500"	Integration Limit (0-32000)
KI	"KI 100"	PID Integration constant (0-32000)
KD	"KD 100"	PID Damping constant (0-32000)
KP	"KP 75"	PID gain (0-32000)
SS	"SS 10"	Single Step jog (1-65,535 counts)
SL	"SL +1000"	Software (travel) Limits ($\pm 10^9$ counts)
UF	"UF"	Update PID filters

PROGRAMMING

ASCII CODE	EXAMPLE	INTERPRETATION AND RANGE
CP	"CP"	Compile Program(s)
DE	"DE 2-7"	Delete program line(s) (1-9999)
EP	"EP"	Enter Program(s)
EX	"EX1"	Execute Program (1-999)
IN	"IN 3 PA+10"	INsert program line (1-9999)
LL	"LL 100-200"	List program Line(s) (1-9999)
LP	"LP 3"	List Program (1-999)
QP	"QP"	Quit PROGRAM EXECUTION mode
/QP	"/QP"	Program delimiter
%	"%"	Quit PROGRAM ENTRY mode
'	" ' "	Program comments

REPORTING

ASCII CODE	EXAMPLE	INTERPRETATION AND RANGE
DA	"DA"	Tell Desired (target) Acceleration
DP	"DP"	Tell Desired (target) servo position
DV	"DV"	Tell Desired (target) servo velocity
MS	"MS"	Tell Motor Status
QS	"QS"	Query Status
TE	"TE"	Tell allowable motor following Error
TF	"TF"	Tell Filters
TL	"TL"	Tell (software) Limits
TM	"TM"	Tell Memory
TP	"TP"	Tell (real) Position

REPORTING (continued)

ASCII CODE	EXAMPLE	INTERPRETATION AND RANGE
TS	"TS"	Tell Status
TV	"TV"	Tell (real) Velocity
VE	"VE"	Tell software Version
?	"?"	Tell contents of error code buffer

SEQUENCE

ASCII CODE	EXAMPLE	INTERPRETATION AND RANGE
DL	"DL LABEL"	Defines Label (A thru Z)
WA	"WA 1000"	WAit (1-65000 milli-secs)
WS	"WS 100"	Wait for Stop
WT	"WT 500"	WaiT before update (100-65000) 10us intervals
WH	"WH LABEL"	While input High...goto...
WL	"WL LABEL"	While input Low...goto...
JL	"JL LABEL 250"	Jump to Label(A thru Z) n(1-255) times
JU	"JU LABEL"	JUmp to label(A thru Z)
ON	"ON +1000 LABEL"	ON...(± 10^9 counts) goto...(label A thru Z)

MISCELLANEOUS

ASCII CODE	EXAMPLE	INTERPRETATION AND RANGE
CB	"CB"	Clear Bit
DI	DI "MESSAGE" 1000	DIsplay message (8 characters, .001-65 sec)
DO	"DO 7"	Do Options (0-255)
FO	"FO 255"	Format Output (0-255)
HE	"HE"	HElp. List DCS750 commands

MISCELLANEOUS (continued)

ASCII CODE	EXAMPLE	INTERPRETATION AND RANGE
HT	"HT"	Select motion status method (0-1)
HX	"HX"	HeX mode
RQ	"RQ 65"	ReQuest for service (64-255)
SB	"SB"	Set Bit
SO	SO 2"	Srq On event (0-8)
SR	SR 7"	Select baud Rate (0-7)
TG	"TG"	ToGgle bit
XX	"XX"	Purge memory & install default parameters

6.3 PROGRAMMING MODES

Essentially, the DCS750 operates in one of three modes:

1 **COMMAND** mode 2 **PROGRAM ENTRY** mode 3 **PROGRAM EXECUTION** mode

In COMMAND mode, commands are executed as they are received. They are not stored in memory. The DCS750 power-up state is COMMAND mode. An example of a sequence of instructions issued in COMMAND mode is shown below.

COMMAND	REMARKS
"1va5000,ac20000"	Set the velocity and acceleration for axis 1
"1dh,pa+100,ws,pa0"	Reset counter, move, wait for stop, and return.

In PROGRAM ENTRY mode a sequence of commands are downloaded from the host computer and stored in non-volatile memory for later execution. Each axis allocates 6k bytes of non-volatile memory for program storage. The "EP" command activates PROGRAM ENTRY mode. Instructions received thereafter are saved in contiguous memory for later evaluation with the "CP" command. To exit PROGRAM ENTRY mode and re-enter COMMAND MODE a percent character, "%", must be issued.

IMPORTANT: SEND A PERCENT CHARACTER (ASCII 37) TO EXIT PROGRAM ENTRY MODE

An example of commands entered in PROGRAM ENTRY mode is shown below.

LINE #	COMMAND	REMARKS
	"3EP"	* Start PROGRAM ENTRY mode (axis #3 only)
0001	"DH"	Define home
0002	"SL +1000"	Positive software travel limit
0003	"SL -1000"	Negative software travel limit
0004	"VA 5000"	Set maximum velocity to 5000 counts/sec
0005	"AC 20000"	Set acceleration/deceleration rate
0006	"DL START"	Define target label for "jump to..."
0007	"PR +500"	Execute relative motion
0008	"WA 1000"	Wait 1 second
0009	"PR -500"	Execute relative motion
0010	"WA 500"	Wait 0.5 second
0011	"JL START 10"	Jump to label "START" ten times
	"%"	Exit PROGRAM ENTRY mode
	"CP"	Compile program
	"EX1"	Execute program 1

IMPORTANT: DO NOT ISSUE CALL NUMBERS TO COMMANDS STORED IN PROGRAM MEMORY.

A DCS750 axis is in PROGRAM EXECUTION mode when it is running a successfully compiled program. It is worth noting that each axis is completely independent with its own program buffer and, therefore, may be placed in PROGRAM EXECUTION mode while another axis remains available in COMMAND mode, for example.

The DCS750 accepts and executes new commands on any axis from either interface, even while running a program. However, several commands like "OR", "DE", "CP", "EX", and "IN", for example, are not accepted by the active axis from the communications link during PROGRAM EXECUTION mode.

Any command or motion errors detected while in PROGRAM EXECUTION mode will cause that axis to abort motion and re-enter COMMAND mode. The axis will automatically re-enter COMMAND mode when the program execution is finished.

IT IS NOT NECESSARY TO IMPLEMENT A CALL NUMBER WITH EACH COMMAND LINE. COMMANDS WITHOUT CALL NUMBERS ASSUME THE LAST NUMBER ISSUED AFTER RESET.

COMMAND DESCRIPTION

DI DIsplay message. The "DI" command permits the user to generate an 8 character message on any axis equipped with the front panel display option. For example, 2DI "TARGET 3" 5000, displays the message "TARGET 3" on axis 2 for 5 seconds. Note that the message within the quotation marks may not exceed 8 characters. Also, the parameter following the second quotation mark is presumed to be the time-delay parameter and may not surpass 65000 (milli-seconds). Note that any *new command received will NOT be executed until after the "DI" time-delay parameter has expired.*

DO Do Options. This instruction allows the programmer to selectively enable/disable various options. i.e., front panel or GPIO jogging, origin search, and/or position resetting. This command permits computer controlled local lock-out emulation. "DO 6", for example, sets bits 1 & 2 to disable local position resetting and origin search activation.

			BIT					
7	6	5	4	3	2	1	0	Use
X	.	.	.	.	.	.	.	0=8MHz, 1=6MHz clock operation
.	X	.	.	.	.	.	.	
.	.	X	.	.	.	.	.	
.	.	.	X	.	.	.	.	1=GPIO output bit monitors origin
.	.	.	.	X	.	.	.	1=GPIO output bit monitors motion
.	.	.	.	.	X	.	.	1=Disable local position reset
.	.	.	.	.	.	X	.	1=Disable local origin search
.	.	.	.	.	.	.	X	1=Disable local jogging

Bit-0 Do not permit front panel or GPIO jogging.
Bit-1 Do not permit front panel or GPIO origin search activation.
Bit-2 Do not permit front panel or GPIO position reset.
Bit-3 If this bit is set the GPIO output bit will go LOW when there is motion and go HIGH (note open collector output) for no motion. Also see "HT" command for bounce/no bounce detection.
Bit-4 If this bit is set (and bit-3 is reset) the GPIO output bit will go LOW when origin search is in progress and go HIGH (note open collector output) when idle.
Bit-7 For factory use only. (Automatically set during reset)

DL Define Label. The "DL" command defines a label (A thru Z) to be used as a target for "JL" and "JU" instructions. Although, only the first character of the label is recognized more are allowed for increased program readability.

DP Desired Position. This command reports the current desired (setpoint) position.

6.4 DETAILED DESCRIPTION OF COMMANDS

COMMAND DESCRIPTION

AB ABort motion. This command loads the present position into the target position register causing the motor to stop *abruptly* and abort further motion. See "ST" command for *soft* stops.

AC ACceleration. The "AC" command sets the trapezoidal acceleration and deceleration portion of the velocity profile generator. The acceleration and deceleration range is from 250 to 10^9 counts/sec^2. *Note that this command may not be changed during motion.*

CB Clear Bit. This instruction clears the programmable output bit low. The open-collector output located on the General Purpose I/O connector is useful in synchronizing external devices to internal events (i.e., motion complete).

CP Compile Program(s). The "CP" command compiles all programs stored in memory. Illegal commands, jumps, labels and parameters stored in memory are evaluated at this time. If one or more errors are detected an error list is generated with line numbers and error type descriptors. If the RS-232-C communication link is active, an error list or "PROGRAM COMPILED" message is transmitted *automatically* (provided an XOFF was not previously received). However, if the GPIB communication link is active the axis will transmit the result of the compilation to the error buffer. And the contents of this buffer is retrieved with the question mark "?" command.

i.e. "CP,?" (compile program & return error status)

DA Desired Acceleration. This command reports the current desired (setpoint) acceleration and deceleration.

DE DElete. The "DE" command deletes specific program lines in memory. The range is from 1 to 9999 program lines. i.e., "DE 10" (delete line 10) or "DE 15-20" (delete lines 15 thru 20) or "DE -100" (delete lines 1 thru 100) or "DE" (delete all program lines). It will be necessary to compile the stored programs after editing. *Note that "DE 0" deletes all program lines!*

DH Define Home. This command sets the position register to absolute zero position and is used to initialize the "home" position of the control axis. To avoid initialization errors, it is important to understand that the current absolute position is merely re-defined to be zero. If there is a difference between the target and the actual position when the "DH" (Define Home) command is executed, such as could be caused by a torque load, then the difference in positions will be *preserved* and appear as an error to an unwary user/observer.

COMMAND DESCRIPTION

FO Format Output. The "FO" instruction permits the user to cut down on the number of characters transmitted with various reporting commands, thus, reducing command response time. The user is encouraged to experiment and obtain the best output for a given application. i.e., "FO 18" (compress header and suppress "END" line).

			BIT					
7	6	5	4	3	2	1	0	Use
X	.	.	.	.	.	.	.	Suppress software limits
.	X	.	.	.	.	.	.	Suppress position display refresh
.	.	X	.	.	.	.	.	Suppress RS232 readability LF
.	.	.	X	.	.	.	.	Suppress "END" line
.	.	.	.	X	.	.	.	Suppress RS232 error message output
.	.	.	.	.	X	.	.	Suppress local speed set & prog run
.	.	.	.	.	.	X	.	Suppress encoder quadrature errors
.	.	.	.	.	.	.	X	Suppress units and long error code

Bit-0 eliminates un-wanted description characters in all reporting commands. i.e, "1>E01" instead of "01> E01 (BAD COMMAND)"

Bit-1 Ignores encoder quadrature signal errors due to transients or poor signal quality. See to section 3.7.

Bit-2 disables front and rear panel double function features. See sections 3.4 and 3.6.

Bit-3 inhibits automatic error message output via full duplex RS-232-C communications link.

Bit-4 suppresses automatic "END" line output for every response. i.e., "01> END" indicating end-of-transmission.

Bit-5 suppresses automatic line-feed output whenever a carriage-return is received to improve RS-232-C terminal readability.

Bit-6 disables position display updating for the purpose of increased overall system speed. The "DISP OFF" message will appear.

Bit-7 disables software travel limits for the purpose of decreased "PR" and "PA" command latency.

HE HElp. The "HE" instruction causes the controller to return a listing of the commands implemented in the current firmware version.

HT Select motion status method. If HT=1 (i.e., "HT1") then "QS","TS", and "SO3" commands report *MOTION COMPLETE* status in its respective bit field. However, if HT=0 (i.e., "HTO") then "QS","TS", and "SO3" commands will report *MOVING/NOT MOVING* status in the motion complete bit field. The "moving/not moving" method is useful for applications which require position disturbance monitoring. Also note that this method may cause multiple GPIB SRQs if "SO3" command is employed.

COMMAND DESCRIPTION

DS Derivative Sampling interval. This command determines the derivative sampling rate (0-255). The derivative signal is multiplied by the coefficient kd (see "KD" command) each derivative sampling interval. This product contributes to the motor control output every sample interval (256us), independent of the user-chosen derivative sampling period. i.e., "DS 0"

BIT								Selected Derivative
7	6	5	4	3	2	1	0	Sampling Interval
0	0	0	0	0	0	0	0	256us
0	0	0	0	0	0	0	1	512us
0	0	0	0	0	0	1	0	1024us
0	0	0	0	0	0	1	1	2048us
1	1	1	1	1	1	1	1	65536us

NOTE: THE 256 CORRESPONDS TO 2048/8MHZ

DV Desired Velocity. This command reports the current desired (setpoint) velocity.

EP Enter Program. The "EP" instruction invokes PROGRAM ENTRY mode. Commands received thereafter are saved in non-volatile memory. In PROGRAM ENTRY mode, the command input channel is redirected from the normal verification and execution routine to character acceptance and storage. *Note that the "%" character (ASCII 37)/carriage* return combination is required to exit this mode and re-enter COMMAND mode.

EX EXecute program. The "EX" command invokes PROGRAM EXECUTION mode. A compiled program (see "CP" command) stored in memory may be executed with this command. Up to 999 programs may exist in each axis. i.e., "3EX2" (execute program 2 on axis 3). While in PROGRAM EXECUTION mode, it will still be possible to enter and execute new commands from either interface. However, any command or motion errors detected will abort the program and re-enter COMMAND mode. (See "/QP" command for multiple program storage)

FE Following Error. This command enables the user to load a threshold (0-32000 counts) for position error. Position error is defined as the difference between the desired position and the actual position. If this error (E17) is reached the motion is stopped by an automatic "motor-off" instruction.

COMMAND DESCRIPTION

HX HeX mode. The "HX" command is used to place an axis into hexadecimal interface mode whereby direct access to the servo system's target position and present position registers is possible. See section 6.6 for detailed description of commands.

IL Integrator Limit. This command allows the user to input a new value (0-32000) for the filter integration-limit coefficient "IL". The integral term of the filter equation is not allowed to exceed this value; the limiting process is useful for preventing integral wind-up. Like all PID filter parameters, the value is not changed until the "UF" (update filter) command is issued.

IN INsert. This instruction inserts commands into specific program lines (1-999). i.e., "IN 4 PA+10". This example inserts command "PA+10" into program line #4. Note: 1) The command previously stored at line #4 is *pushed* onto line #5 and so on. 2) The program will must be compiled again before execution is allowed.

IP Invert Phase. The "IP" command inverts the loop feedback phasing. This command is useful in debugging none-factory motor/positioner and/or encoder hook-ups. i.e., "IP 1" inverts standard phasing and "IP 0" restores standard phasing. *Note that applications involving non-standard phasing may affect hardware end-of-travel logic.*

JL Jump to Label. This command uses the first letter of a label as a jump to target and the second argument (1-255) as the repetition count. i.e., "JL LABEL 5". Each time this instruction is encountered the count is decremented by one. If the count is a non-zero after the decrement, program execution will jump to the specified label. When the count reaches zero, program execution continues with the instruction immediately following the "JL" command. *Note that if the repetition argument is zero or not specified the axis will loop indefinitely!*

JU JUmp. The "JU" command is very similar to the "JL" command. However, this instruction is used to loop indefinitely and does not require a repetition command. This command also uses the first letter of a label as a jump to target.

KD Derivative constant. This command allows the user to input a new value (0 to 32000) for the filter derivative (high-pass) coefficient KD. Like all other PID filter parameters, the value is not changed until the "UF" (Update Filter) command is issued. (Also see "DS" command)

COMMAND DESCRIPTION

KI Integrator constant. This command allows the user to input a new value (0 to 32000) for the filter integral (low-pass) coefficient KI. Like all other PID filter parameters, the value is not changed until the "UF" (Update Filter) command is issued.

KP Proportional constant. This command allows the user to input a new value (0 to 32000) for the filter proportional (gain) coefficient KP. Like all other PID filter parameters, the value is not changed until the "UF" (Update Filter) command is issued.

LL List Lines. This command outputs a listing of the currently stored program(s) lines. The range is from 1 to 9999 program lines. i.e., "LL 10" (list line 10) or "LL 15-20" (List lines 15 thru 20) or "LL-100" (List lines 1 thru 100).

LP List Program. This instruction outputs a listing of the currently stored program(s). The range is from 1 to 999 programs. i.e., "LP 2" (list program 2). Essentially, this instruction uses "/QP" commands to determine the starting and ending lines of the program to be listed.

MF Motor oFf. This command stops the motor with zero volts for the motor command signal. This produces braking proportional to velocity (caused by motor back-emf). No holding torque is provided after the motor has stopped.

MO Motor On. The "MO" command restores power to the motor and provides holding torque if required. This command is the inverse of the "MF" instruction.

MS Motor Status. The "MS" command reports the motor status. i.e., "02> 129" (encoder quadrature error). The "TS" command is a non-coded version of the "MS" and "QS" commands. Try it.

			BIT					
7	6	5	4	3	2	1	0	Use
X	.	.	.	.	.	.	.	Encoder quadrature error
.	X	.	.	.	.	.	.	Motor following error
.	.	X	.	.	.	.	.	Positive travel software limit
.	.	.	X	.	.	.	.	Negative travel software limit
.	.	.	.	X	.	.	.	Positive travel hardware limit
.	.	.	.	.	X	.	.	Negative travel hardware limit
.	.	.	.	.	.	X	.	Motor off
.	.	.	.	.	.	.	X	Motor error (bits 2-7 ORed)

COMMAND DESCRIPTION

MV MoVe. This command places the axis in the velocity mode of operation. If the command is followed with a "-" sign (i.e., "MV-"), the positioner will move indefinitely (or until a limit is encountered) in the negative direction and vise-versa for "MV+" or "MV". The velocity at which the axis will move is the *last* velocity command level entered. When operating in the velocity mode, the motor accelerates to the specified velocity at the specified velocity until commanded to stop. The velocity is maintain by advancing the desired position at a constant rate. If there are any disturbances to the motion during velocity mode operation, the long-time average velocity remains constant. *Note that the axis does not wait before proceeding to the next command during PROGRAM EXECUTION mode like with the "PA" or "PR" commands.*

OA Origin Acceleration. The "OA" command sets the acceleration and deceleration portion of the velocity profile generator for front panel jog and origin search positioning. The acceleration and deceleration range is from 250 to 10^9 counts/sec^2. *Note that this command may not be changed during motion.*

OH Origin High speed. This command allows the user to input a new (target) value (0-1,000,000 counts/sec) for absolute velocity for high speed front panel jog and origin search positioning. The "OH" parameter may also be entered via front panel high speed setting.

OL Origin Low speed. This command allows the user to input a new (target) value (0-1,000,000 counts/sec) for absolute velocity for low speed front panel jog and origin search positioning. The "OL" parameter may also be entered by font panel low speed setting.

ON ON count. This instruction generates an internal and/or external interrupt whenever the specified incremental trip point (±1,000,000,000 counts) has been reached. The internal interrupt redirects the program flow from its normal progression, to the specified label (see "DL" command). If the "SO" command is active with option 8, an IEEE-488 SRQ is automatically asserted with a fixed status byte of 255. "ON 0" disables further interrupts. Below is a sample program using both "ON" and "RQ" commands.

```
"1EP"
"ON +1000 SERVICE       'GENERATE SRQ EVERY +1000 INCREMENTS"
"DL THERE               'TARGET LABEL"
"PA +8000               'MOVE TO ABSOLUTE POSITION +8000"
"QP                     'QUIT PROGRAM EXECUTION MODE"
"DL SERVICE             'PROGRAM REDIRECTED HERE"
"RQ 96                  'ASSERT GPIB SRQ WITH STATUS 96"
"JL THERE 7             'CONTINUE TOWARD POSITION +8000"
"ON 0                   'DISABLE SRQ GENERATION AFTER +7000"
"JU THERE               'CONTINUE TOWARD POSITION +8000"
"%"
```

COMMAND   DESCRIPTION

OR   ORigin search. This command invokes the Klinger origin search algorithm which automatically seeks a predetermined *home* or origin. The algorithm will locate the optional mechanical, electro-optical or electro-magnetic origin *switch* located within the positioner when the "OR 1" or "OR 2" command is received. The "OR 1" option is used for Klinger standard switch origin search. However, the "OR 2" option is used for non-standard switch origin searches. "OR 0" or "OR" will cause the positioner to execute a *floating* origin search, or move to location zero (+0). In either a *floating* or switch origin search the positioner will travel at the speed and acceleration stored with the OA, OH, and OL commands. The origin search switch and *floating* algorithms are designed to reach target position quickly and reduce mechanical backlash. Note: The OH and OL commands may also be set via front panel jog speed setting procedures.

PA   Position Absolute. The "PA" command declares the absolute position mode of operation. The argument (0 to $\pm 10^9$ counts) is treated as the target position relative to position zero. The "DH" command is used to define the zero position. The direction of motion and number of steps to be taken are automatically calculated and executed. *Note: 1) The "PA" instruction will automatically "turn-on" a previously "turned-off"* motor. *2) This command may be executed "on the fly".*

PR   Position Relative. The "PR" command declares the relative position mode of operation. The argument (0 to $\pm 10^9$ counts) is treated as the number of counts to be taken from the present absolute position in the specified direction. *Note that the "PR" instruction will automatically "turn-on" a previously "turned-off" motor.*

QP   Quit Program. The "QP" command causes the axis to quit PROGRAM EXECUTION mode and re-enter COMMAND mode. In conjunction with the "WH" or "WL" commands, for example, this instruction is useful in programming conditional program breaks. This command may be issued at anytime through either communication link to facilitate a real-time program break.

/QP   /Quit Program. This is command is used to separate contiguous programs in memory and quit further program execution. This instruction is used as a program delimiter. Up to 999 may exist in each axis. Below are 2 one line programs stored in axis #3.

```
"3EP"
"PA +1000            'PROGRAM 1"
"/QP"
"PA +3000            'PROGRAM 2"
"%"
```

Note that the "%" (exit PROGRAM ENTRY mode) command automatically generates a "/QP" after the last program line. Also note that "/QP" commands may be inserted (see "IN" command) or deleted (see "DE" command as needed to separate or combine programs.

COMMAND **DESCRIPTION**

SO Srq On event. This command asserts the GPIB Service Request (SRQ) on a selected occurrence. i.e., "SO 4" (generate SRQ on completed timer). See sections 5.4 and 5.5 before using this command. Also see "ON" command for option 8.

OPTION		OCCURRENCE GENERATOR
* 0	◄--------►	DISABLE SRQ FEATURE (default)
1	◄--------►	MOTOR ERROR OR COMPLETED ORIGIN
2	◄--------►	MOTOR ERROR OR COMPLETED PROGRAM
3	◄--------►	MOTOR ERROR OR COMPLETED MOTION
4	◄--------►	COMPLETED TIMER (see "WA" or "WS" command)
5	◄--------►	HIGH-TO-LOW TRANSITION ON GPIO INPUT BIT
6	◄--------►	LOW-TO-HIGH TRANSITION ON GPIO INPUT BIT
7	◄--------►	MOTOR ERROR (see "MS" command)
8	◄--------►	POSITION TRIP POINT (see "ON" command)

SR Set Rate. This command sets the RS232 baud rate. i.e., "SR 7" (9600 baud) Any changes in baud rate will not take affect until after a system reset or power cycle. The DCS750 serial interface is always configured for *8 data bits, 1 stop bit, and no parity.*

BAUD RATE		PARAMETER
110	◄--------►	0
150	◄--------►	1
300	◄--------►	2
600	◄--------►	3
1200	◄--------►	4
2400	◄--------►	5
4800	◄--------►	6
9600	◄--------►	7 (default)

NOTE: ALL AXES SHOULD BE SET TO THE SAME BAUD RATE BEFORE A RESET OR POWER CYCLE.

SS Single Step. This command allows the user to specify the number of encoder counts to increment/decrement when "single stepping" the positioner via local jog switches. i.e., "SS 10", causes the positioner to increment/decrement 10 encoder counts each time the front panel jog switch is momentarily depressed. The range is from 1 to 65,535.

ST STop motion. This command halts the positioner/motor using programmed deceleration (see "AC" command). Whereas, the "AB" command performs a *hard* stop.

TE Tell Error. This command reports the current "FE" (following error) parameter in use.

COMMAND DESCRIPTION

QS Query Status. The "QS" command reports general axis status. i.e, "01> 64" (local jog mode active). Also see "TS" command.

			BIT					
7	6	5	4	3	2	1	0	Use
X	.	.	.	.	.	.	.	Origin Search active
.	X	.	.	.	.	.	.	Local jog mode active
.	.	X	.	.	.	.	.	Velocity mode active
.	.	.	X	.	.	.	.	GPIO input bit LOW
.	.	.	.	X	.	.	.	Timer active
.	.	.	.	.	X	.	.	Program active
.	.	.	.	.	.	X	.	Motor error
.	.	.	.	.	.	.	X	Motion complete (see "HT" command)

RQ ReQuest service. This command asserts the IEEE-488 (GPIB) SRQ line and supports a programmable status byte for the GPIB serial poll. The byte range, however, is from 64 to 255. Bit 6 is always set in accordance with the protocol. i.e., "RQ65" Only a GPIB serial poll or system reset will release the SRQ line after it has been asserted. With this command the user may efficiently signal the host computer as to the level or progress of the program flow.

SB Set Bit. This instruction sets the programmable output bit high. The open-collector output located on the General Purpose I/O connector is useful in synchronizing external devices to internal events (i.e., motion complete).

SL Software Limit. The "SL" command sets the software travel limits ($\pm 10^9$ counts) relative to position zero. i.e., "SL+10000" and "SL-7000". This sets the positive software limit to position +10000 and the negative limit to position -7000. Now, if the programmer inadvertently instructs the controller to move the positioner beyond the software limits, the motion is automatically aborted at the set limit. *Note: 1) The "DH" commands affects the software limits. 2) "PR" commands entered "on-the-fly" may affect the software limit function also. 3) Both, positive and negative limits are entered with the same "SL" command. 4) "FO" command bit 7 disables both software limits and reduces "PA" latency.*

The "LIMIT" message is momentarily shown on the position display indicating a software and/or hardware travel limit was encountered.

COMMAND DESCRIPTION

TF Tell Filters. This instruction reports the current PID filter parameters in use.
For example; "01> KP=75 KD=100 KI=100 IL=100 DS=0"

The "TF" command always reports the PID filters entered prior to the "UF" (update filters) instruction because only those parameters are currently implemented in the servo system.

TG ToGgle bit. This command generates a 50us *low going* pulse on the GPIO programmable (open collector) output bit. This instruction is functionally equivalent to successive "CB" "SB" commands, however, the pulse width is shorter and more accurately controlled.

TL Tell Limit. This instruction reports the current "SL" parameters in use.
For example; "01> SL=+1000 SL=-1000"

TM Tell Memory. This command reports the program storage memory usage.
For example; "02> 10 BYTES USED 3090 BYTES FREE"

TP Tell Position. The "TP" command reports the *real* location of the position.

TS Tell Status. The "TS" command reports axis status and motor errors (if any).

```
i.e., "01> MOTOR..........ON"
      "01> POSITIONER.....MOVING"
      "01> SYSTEM MODE....COMMAND"
        *
      "01> END"
```

TV Tell Velocity. This command reports the *real* velocity of the positioner. However, to get an accurate reading the positioner must be in continuous motion (not accelerating or decelerating) for a minimum of 1 second.

UF Update Filters. This command transfers new filter parameters from the input registers to the working registers, thereby effecting the change in loop compensation filter tuning. This fact can be used advantageously; the user can input numerous PID data ahead of their actual use and this simple *pipeline* effect can relieve potential host controller data communications bottlenecks.

COMMAND DESCRIPTION

VE tell VErsion. This command reports the revision level of the firmware in the ROM in use.

```
02>
02>      KLINGER SCIENTIFIC CORPORATION
02>          DCS750 MOTOR CONTROLLER
02>               VERSION x.xx
02>
02> END
```

VA Velocity Absolute. This command allows the user to input a new (target) value (0-1,000,000 counts/sec) for absolute velocity. Note: The velocity may be changed *on-the-fly*.

VR Velocity Relative. This command allows the user to input a new value (±1,000,000 counts/sec) for relative velocity. The new velocity will become the old target plus the new relative value. (i.e., VA=VA+VR) *Note that the final value of velocity (the old target value plus the new) must remain positive and within the 0 to 750,000 counts/second range.*

VS Sample time Velocity. This instruction allows the user to input a new (target) velocity value (0-16,777,778) for absolute velocity. The "VS" command parameter is in terms of counts/65536/256us (sample time), <u>not</u> counts/second. This instruction is recommended for applications requiring faster velocity updating than can be accomplished with the "VA" and "VR" commands.

WA WAit. The "WA" instruction will time-delay further <u>program</u> execution for the number of milliseconds (0-65000) specified. i.e., "WA1000" (waits 1 second).

WH While High. This command tests the General Purpose I/O connector input bit (see Chapter 8) for a <u>high</u> and branches to the specified label (A thru Z) if true. Otherwise, program execution continues with the next command. i.e., "WH LABEL" (branches to "DL LABEL" if input bit high). The "WH" instruction is useful in synchronizing internal program execution to external events.

WL While Low. This command tests the General Purpose I/O connector input bit (see Chapter 8) for a <u>low</u> and branches to the specified label (A thru Z) if true. Otherwise, program execution continues with the next command. i.e., "WH LABEL" (branches to "DL LABEL" if input bit low). The "WL" instruction is useful in synchronizing internal program execution to external events.

COMMAND DESCRIPTION

VS Wait for Stop. This instruction causes the axis to wait until the motor has virtually stopped moving for the specified time (0-65000 ms) before proceeding to the next command. It is most useful in COMMAND mode where successive motion commands may overwrite previous motion commands not yet completed. i.e., "1PA100,WS1000,PA0" (move, wait one second then move again). *Note that in PROGRAM EXECUTION mode each axis automatically waits for the motion or sequence type command to be completed before proceeding to next instruction.*

WT WaiT before update. This command in conjunction with the "VS" command facilitate velocity contouring. The "VS" command will not update the new velocity parameter until the DCS750 free running timer previously loaded with the "WT" parameter has expired. Once the time interval has expired the "VS" parameter is then processed and the timer is automatically reloaded with the same value. This processed will continue until a "WT0" is encountered or program execution is terminated. The parameter range is from 100 to 65000 in 10uS increments. i.e., "WT200" causes 2msec update intervals. This command is typically setup and processed only once. Therefore it should be located "early" in your program. Note that when this command has been processed the DCS750 will not wait for a "PA" or "PR" instruction to be completed before proceeding the next instruction as would normally happen. Instead, the processor will only pause at "VS" commands for as long as specified by the "WT" parameter. In addition, while in this mode no other timer related commands may be used (i.e., "WA" and "WS"). Please refer to the sample programs in Chapter 7 implementing "WT" commands.

```
EP                      'Enter Program Mode
OR1,WA1000,DH           'Define origin switch position as home
AC 75798                'Minimum required acceleration
WT 1000                 'Time interval = 10ms
DL A                    'Target label A
PA+ 24000               'Total displacement
DL B                    'Target label B
VS0                     '1st velocity update
VS16940                 '2nd velocity update
VS33857
VS50774
VS67647
VS50774
VS33857
VS16940
PA0                     'Move back to starting location
JL B 2                  'Repeat velocity update sequence
JL A 10                 'Repeat entire cycle 10 times
%                       'End of program
```

XX This command is used to purge the contents of non-volatile memory. All previously stored programs, trajectory parameters, and PID coefficients are erased and replaced with default values. Consult with section 6.5 for PID system compensation.

% Percent (ASCII 37). The percent character causes the axis to exit PROGRAM ENTRY mode and re-enter COMMAND mode.

? Question mark (ASCII 63). Via the RS-232C error messages are automatically transmitted to the host (unless supressed by "FO" command bit-3). However, the IEEE-488 (GPIB) is a half-duplex interface and the "?" command must first be received and then the talk address can be asserted to get the error message (if any).

```
i.e.,       "?"                (query error buffer)
            EOO NO ERROR       (no errors occurred)
```

* Asterisk (ASCII 42). Immediately after the asterisk character is received from either IEEE-488 or RS-232-C communications link, the axis is forced into a "warm" reset state. Program execution (if active) is halted, the motor is turned-off and internal varaibles are cleared. However, unlike a "cold" reset, the servo controller is <u>not</u> reset and the position counter is preserved.

, Comma (ASCII 44). The comma is used to separate multiple commands transmitted on a single line to the <u>same</u> axis. Line execution begins when a carriage-return is received.
i.e., "1PA+100,WS200,PA0,WS500,TP"

; Semicolon (ASCII 59). The semicolon is also used to separate multiple commands transmitted on a single line to the <u>same</u> axis. Line execution begins when a carriage-return is received.
i.e., "1PA+200;WS300;PA0;WS500;TP"

' Apostrophe (ASCII 39). Characters received after an apostrophe are not evaluated during compilation to facilitate program line comments during PROGRAM ENTRY mode.

```
i.e.,       "EP"
            "PA+2000   'FIRST TARGET"
            "PA-1500   'SECOND TARGET"
            "%"
```

6.5 SYSTEM PID COMPENSATION

IMPORTANT

If you need to compensate standard Klinger positioners please proceed to section 6.6, Suggested Compensation Parameters. System compensation is typically performed at the factory. The axis should be properly configured and compensated for the positioner in use. The positioner motor assembly and DCS750 rear panel motor connector(s) should also have matching labels for easy identification and pairing. The suggested operating PID and trajectory parameters are stored in non-volatile memory and are automatically reloaded after a reset or power-cycle.

Assuming there is no loop inversion problem, or that one has been detected and corrected (see "IP" command), the loop compensation filter may now be tuned. The axis should be in the following condition;

```
KP=1    KD=0    KI=0    IL=0    DS=0
```

where, the motor is holding position, but with only slight available torque.

The first step is to only increase the filter proportional (gain) parameter (KP) in small steps until the loop gets stiffer, but without oscillation. Begin with the following commands:

```
KP 2    (increase KP by 1)
UF      (update filter tuning)
```

Does the motor feel noticeably stiffer? Test the damping visually (if possible) to determine if there is a tendency to ring. This can be seen by forcing the motor away from its desired holding position and suddenly releasing it. Observe its settling behavior. Continue in this manner, perhaps doubling the "KP" parameter with each step, until ringing (but not oscillation) is visible. Marking the motor shaft with a well-defined pencil line on a light-colored tape background helps visualize the effect. Remember, the command "UF" must be issued after entering each new "KP" value. Stop this iterative process when the motor settling rings noticeably (such that there is reason to fear that another doubling of "KP" might lead to oscillation). If violent oscillation is encountered, remember that command "MF" removes power from the motor (so does the power switch!).

The next step is to squelch the ringing without sacrificing the system stiffness that has been acquired. This is done by increasing the value of the filter derivative parameter KD (Also see "DS" command) and/or increasing the contribution of the tachometer feedback (if any) on the amplifier potentiometer, P1. Like the "KP" parameter, this is also done in steps. The goal is to eliminate the ringing during settling without introducing system oscillation. Begin with the following commands:

```
KD 1  (KD was previously 0)
UF    (update filter tuning)
```

Is the time or amplitude of the ringing noticeably less? Test ringing behavior visually while iteratively increasing the "KD" parameter with each step until ringing is no longer visible. Remember to issue the "UF" command after each new value of "KD". Do not increase "KD" so much as to introduce oscillation. Keep command "MF" in mind for stopping motion. Oscillation caused by an excessive "KD" value is typically a much higher frequency than that caused by too high a "KP" value.

Repeat the entire above process for tuning "KP" and "KD" (continuing from the presently determined parameter values), but use smaller parameter-value steps this time.

The axis should now be stable. This assumes that the user will not be subjecting the system to varying inertial loads. Also assumed for now is that the motor will not be working against a steady state torque load. This can cause a position error of a few encoder counts, which can be eliminated by using the filter integral parameter (KI) and its related integration-limit parameter (IL).

To verify that the system is now operational, enter the following commands:

```
DH          (Define home)
VA 5000     (5000 encoder counts/second)
AC 20000    (20000 encoder counts/second/second)
PA 500      (500 encoder counts)
```

then issue the "TP" (tell position) command and notice that the current position is within a few counts of position 500. Move the positioner/motor back to its origin with the command:

```
PA 0        (Position absolute 0)
```

After the move, issue the "TP" command to show that the motor is within a few counts of position 0.

TUNING FILTER COEFFICIENTS KI AND IL

Coefficient "KI" is the integral, or low-pass, coefficient of the PID control algorithm. Its chief contribution is added system DC and low-frequency gain, which helps resist a constant torque static load (such as loading by gravity). Because its effect is accumulative with time, it must sometimes be limited to prevent what is called integral (or reset) *wind-up*, a backlash effect. Coefficients "IL", or integration limit, is provided for this purpose. The value loaded for this parameter acts as a clamping value on the effect of coefficient "KI", and prevents further integration of the error term. Coefficient "KI" has no effect if the value of "IL" is zero; so when attempting to tune "KI" be sure to first start with some small value for coefficient "IL".

The value for "KI" should be as small as is consistent with eliminating the positional error effect caused by the static torque loading. This becomes apparent when one realizes that "KI" introduces another pole (more phase lag) into the loop response. The need for using "KI" can be determined by reading the real position (TP) and comparing it to the desired position (DP). If there is a problem with static loading there will be a position error after making a move. Careful tuning of coefficients "KI" and "IL" will eliminate this error.

6.6 SUGGESTED PID COMPENSATION PARAMETERS

Although there are many exceptional applications with unusual motor/load characteristics requiring unique PID parameters, most applications seemingly incorporate relatively common Klinger positioner configurations. Therefore, we were able to compile a list of PID parameters for various Klinger positioners.

NOTE THAT THE FOLLOWING ARE MERELY SUGGESTED PID PARAMETERS ONLY AND MAY NOT BE SATISFACTORY FOR YOUR POSITIONER AND/OR APPLICATION.

POSITIONER	PID PARAMETERS					NOTES			
GV88........	KP200	KD10000	KI100	IL100	DS0	2000ppr	encoder,	UE72CC	motor
MT160.......	KP150	KD200	KI50	IL50	DS0	"	"	"	"
RT300.......	KP200	KD5000	KI100	IL100	DS0	"	"	"	"
RT120.......	KP100	KD500	KI200	IL200	DS0	"	"	UE42CC	"
UT100.......	KP100	KD500	KI200	IL200	DS0	"	"	"	"
UR100.......	KP100	KD500	KI200	IL200	DS0	"	"	"	"
UR100.......	KP1500	KD10000	KI200	IL200	DS0			UE30CC	"
UT100.......	KP1500	KD10000	KI200	IL200	DS0			"	"
BMCC........	KP1500	KD10000	KI400	IL400	DS0			UE22CC	"
MT160HD.....	KP11	KD0	KI200	IL30	DS0	Linear encoder w/tachometer			

6.7 HEXADECIMAL INTERFACE MODE

COMMAND: HX...HeX mode (IEEE-488 only)

FUNCTION: The "HX" command is used to place an axis into hexadecimal interface mode (HEX mode) whereby direct access to the servo system's target and present position registers is possible. In this mode the command interpreter, which is responsible for converting ASCII instructions into hexadecimal, evaluating syntax and parameter range, is bypassed.

The Hex mode environment provides the following advantages:

1) Fast new target position updating.
2) Real-time position status reporting.

While in HEX mode, position display updating and pushbutton recognition are disabled. The DCS750 will also display the message "HEX MODE" on each activated axis.

FORMAT: The DCS750 has a 32-bit (4 byte) target position register. Therefore, four bytes plus a Carriage-Return (0Dh) are required for proper IEEE-488 communication. The position register accepts a 4 byte signed hexadecimal value between C0000000h and 3FFFFFFFh. For example, to get to position -1 (decimal) the user has to transmit FFFFFFFFh plus a Carriage-Return to the target axis. Out of range parameters are simply ignored.

BYTE 1	BYTE 2	BYTE 3	BYTE 4	BYTE 5
(MSByte)			(LSByte)	(0Dh)

The DCS750 also has a 32-bit (4 byte) encoder position register. Four hexadecimal position bytes (MSByte first) plus a Carriage-Return (0Dh) are automatically output whenever the axis IEEE-488 TALK address is placed on the bus. Simply put, if an axis detects its LISTEN address on the bus it will accept 5 bytes. If the TALK address appears, the axis will output 5 bytes.

EXAMPLE: Say the user wanted to move to absolute encoder position 65535 (decimal). The following sequence must occur:

<u>STEP</u>

1) The host computer places the axis IEEE-488 LISTEN address on the bus.
2) The host outputs 00h. i.e., OUTPUT CHR$(0)
3) The host outputs 00h. i.e., OUTPUT CHR$(0)
4) The host outputs FFh. i.e., OUTPUT CHR$(255)
5) The host outputs FFh. i.e., OUTPUT CHR$(255)
6) The host outputs 0Dh. i.e., OUTPUT CHR$(13)

Note that the five bytes are normally concatenated with one "OUTPUT" function. i.e., OUTPUT CHR$(0)+CHR$(0)+CHR$(255)+CHR$(255)+CHR$(13).

ERROR: In the event of a time-out or premature break in the 5 byte transmission, to or from the target axis, an IEEE-488 Service ReQuest (SRQ) interrupt will be asserted to alert the host computer. The host computer is then expected to perform a Serial Poll on those axes capable of causing the interrupt. The axis returning a status byte of 40h during an IEEE-488 Serial Poll is assumed to be the culprit.

EXIT: There are three(3) methods of exiting the HEX mode environment:

1) <u>Turn the system off!</u> The DCS750 always returns to the normal command interpreter after a power-cycle.
2) <u>Perform a system reset!</u> Depress the (RED) RESET switch located just above the POWER switch. This will cause all axes to reboot.
3) Send four dummy bytes plus a non-Carriage-Return 5th byte. A non-Carriage-Return causes the axis to exit HEX mode and re-enter COMMAND mode without rebooting the system. This method preserves the last target position.

NOTE: To reduce command latency to an absolute minimum the firmware does not monitor all nor inform the user of any positioning errors. For example, software limits and encoder quadrature errors are not monitored in this environment. However, the system does recognize hardware limits and following errors although they are not reported.

The firmware uses the last following error trip point, acceleration and velocity parameters loaded into its registers in COMMAND mode to execute new target moves in HEX mode.

HINT: Learn the positioner/load characteristics in the normal COMMAND mode environment first. Make sure that the following error, velocity, acceleration, and displacement parameters are realistic and safe before operating in HEX mode.

TIMING: In HEX mode, any axis engaged in an IEEE-488 TALK or LISTEN byte transfer will time-out 500 milli-seconds after a premature break in transmission.

The read/write latency (*from the time the TALK or LISTEN* address is placed on the *IEEE-488 bus, and all five(5) bytes have been* received/transmitted, to the time the *same axis is ready* to receive/transmit again) will vary depending on the speed of the host computer and its IEEE-488 controller interface card. However, command latency values are provided, using an IBM-AT (10 MHZ) and National Instruments GPIB controller, just for reference.

Real position read latency = 1.0 milli-second
Target position write latency = 0.6 milli-second

CHAPTER 7
SAMPLE PROGRAMS

The following is an interactive sample program using a Real Time Devices Inc IEEE-488 interface for the IBM-PC.

REAL TIME DEVICES

```
1 ' IMPORTANT:   Set axis 1, 2, 3 and 4 to address 1, 2, 3 and 4.
2 '
3 ' *****************************************************************
4 '  Lines 10-190 must be included in front of the application program.
5 '  These lines are copied from the GPINT.bas program supplied on the
6 '  distribution diskette by Real Time Devices, (C) Copyright 1985.
7 ' *****************************************************************
8 '
9 '
10 DEFINT A-Z
20 DEF SEG = 0 : LIBSEG = VAL("&H" + HEX$(PEEK(1018) + PEEK(1019) * 256))
30 FOR LIB = 1 TO 1
40
50
60
70
80
90
100
110
120
130 IF GP100 = 0 THEN PRINT "Missing GP100 Subroutine Library" : end
140 ' Initialize offsets
150 INIT1 = 256
160 CALL INIT1(INIT2,INIT3,INIT4,INIT5,VERIFY,VERSION,GPRESET,GPNEW,GPTIMER,
                        MESSAGE,TEXT)
170 CALL INIT2(WBYTE,RBYTE,WWORD,RWORD,WDWORD,RDWORD,WQWORD,RQWORD,WSTR,
                        RSTR,WARRAY,RARRAY,BWRITE,BREAD,WDMA,RDMA,DMA,TERM,
                        NOTERM,CRLF,EOI,PARSER)
180 CALL INIT3(CONTROL,IFC,REN,RENCLR,RENCLR,RENLOC,REMOTE,TCS,TCA,STANDBY,
                        GPGET,TRIGGER)
190 CALL INIT4(UNLISTEN,UNTALK,MYLISTEN,LISTEN,XLISTEN,ALISTEN,TALK,
                        XTALK,BUSCOM,PASS,SPOLL,APOLL,REQUEST,PPOLL,CONFIGURE,
                        RESPONSE,MYSTATUS)
200 '
```

```
210 ' ***************** APPLICATION PROGRAM INITIALIZATION ****************
220 '
230 COUNT = 100 : CR$ = CHR$(13) : DEVICE = 1 : TEMP = 0 : CLS
240 CALL GPRESET(STATUS) : CALL IFC(STATUS) : CALL GPTIMER(COUNT)
250 CALL NOTERM : CALL UNLISTEN(STATUS) : CALL UNTALK(STATUS)
260 TR$=CHR$(10): SELECT= -1: CALL PARSOR(SELECT,TR$)
270 '
280 ' ************************* ENTER COMMAND *******************************
290 '
300 INPUT "Enter Command..." ; CMD$
310 TEMP = VAL(CMD$)   'axis address (i.e., "3TP")
320 IF TEMP = 0 THEN 330 ELSE DEVICE = TEMP:CMD$=RIGHT$(CMD$,(LEN(CMD$)-1))
330 CMD$=CMD$ + CR$   ' terminate command with carriage return
340 GOSUB 400         ' send command to DCS750
350 GOSUB 500         ' get string (if any)
360 GOTO 300          ' repeat process
370 '
380 '
400 ' ********************** SEND COMMAND TO DEVICE ************************
410 '
420 CALL UNLISTEN(STATUS)
430 CALL LISTEN(DEVICE,STATUS) : CALL MYTALK(STATUS)
440 CALL WSTR(CMD$,BYTES,STATUS) : CALL UNLISTEN(STATUS):CALL UNTALK(STATUS)
450 RETURN
460 '
500 ' *************************** GET STRING ********************************
510 '
520 Q$=STRING$(80," ")
530 CALL UNLISTEN(STATUS) : CALL UNTALK(STATUS) : CALL MYLISTEN(STATUS)
540 CALL TALK(DEVICE,STATUS)
550 CALL RSTR(Q$,BYTES,STATUS)
560 CALL UNTALK(STATUS) : CALL UNLISTEN(STATUS)
565 PRINT $Q                 'print to screen
570 CHKEND$ = MID$(Q$,5,3)   'check for "END" (i.e., "02> END")
575 IF CHKEND$ <> "END" THEN 520 ELSE RETURN
580 '
590 END
```

The following is a another interactive terminal program this time using the National Instruments Inc. PCII IEEE-488 interface for the IBM-PC.

NATIONAL INSTRUMENTS

```
10 '
20 ' IMPORTANT:   Set axis 1, 2, 3 and 4 the address 1, 2, 3 and 4.
30 '
40 ' *********************************************************************
45 '
50 ' Lines 90 to 140 must be included in the front of your application
55 ' program.  These lines are copied from the DECL.BAS program supplied
60 ' on the distribution diskette by National Instruments,
65 ' (C) COPYRIGHT 1984, 1985
70 '
75 '**********************************************************************
80 '
90          CLEAR     ,59000!       ' Basic Declarations
100         IBINIT1 = 59000!
110         IBINIT2 = IBINIT1 + 3
120         BLOAD "bib.m",IBINIT1
130         CALL IBINIT1(IBFIND,IBTRG,IBCLR,IBPCT,IBSIC,IBLOC,IBPPC,IBBNA,
                               IBONL,IBSRC,IBSRE,IBRSV,IBPAD,IBSAD,IBIST,
                               IBDMA,IBEOS,IBTMO,IBEOT,IBRDF,IBWRTF)
140         CALL IBINIT2(IBGTS,IBCAC,IBWAIT,IBPOKE,IBWRT,IBWRTA,IBCMD,IBCMDA,
                               IBRD,IBRDA,IBSTOP,IBRPP,IBRSP,IBDIAG,IBXTRC,
                               IBRDI,IBWRTI,IBRDIA,IBWRTIA,IBSTA%,IBERR%,IBCNT%)
150 '
160 ' **********************************************************************
170 '
180 BDNAME$="DEV1" : CALL IBFIND (BDNAME$,AXIS1%)
185 BDNAME$="DEV2" : CALL IBFIND (BDNAME$,AXIS2%)
190 BDNAME$="DEV3" : CALL IBFIND (BDNAME$,AXIS3%)
195 BDNAME$="DEV4" : CALL IBFIND (BDNAME$,AXIS4%)
200 '
210 ' The label -- DEV1 -- is assigned to address 1 in the
220 ' configuration program: -- IBCONF --
230 '
240 ' ***********************************************************************
250 '
260 ' Set and enable read string terminator
270 ' Note: This can also be done in config. program  -- IBCONF --
280 '
290 EOSV%=&HA                  ' Input string terminator is line feed
300 V%=EOSV%+&H400             ' 0400 means to terminate when EOS is received
310 CALL IBEOS (AXIS1%,V%): CALL IBEOS (AXIS2%,V%)
320 CALL IBEOS (AXIS3%,V%): CALL IBEOS (AXIS4%,V%)
330 CR$=CHR$(13) : DEVICE =1 : TEMP = 0
```

```
370 INPUT "Enter Command" ; CMD$
380 TEMP = VAL(CMD$)       ' address prefix (i.e., "3MO")
390 IF TEMP = 0 THEN 400 ELSE DEVICE = TEMP:CMD$=RIGHT$(CMD$,(LEN(CMD$)-1))
400 CMD$ = CMD$ + CR$
410 GOSUB 450              ' send command
415 GOSUB 540              ' get string
420 GOTO 370
430 ' ************************* SEND COMMAND ******************************
440 '
450 IF DEVICE = 1 THEN CALL IBWRT(AXIS1%,CMD$):RETURN
460 IF DEVICE = 2 THEN CALL IBWRT(AXIS2%,CMD$):RETURN
470 IF DEVICE = 3 THEN CALL IBWRT(AXIS3%,CMD$):RETURN
480 CALL IBWRT(AXIS4%,CMD$):RETURN
510 '
520 ' ************************* GET STRING *********************************
530 '
540 FETCH$ = SPACE$(255)      ' read until 255 bytes or terminator is received
550 IF DEVICE = 1 THEN CALL IBRD(AXIS1%,FETCH$):GOTO 590
560 IF DEVICE = 2 THEN CALL IBRD(AXIS2%,FETCH$):GOTO 590
570 IF DEVICE = 3 THEN CALL IBRD(AXIS3%,FETCH$):GOTO 590
580 CALL IBRD(AXIS4%,FETCH$)
590 FETCH$ = LEFT$(FETCH$,IBCNT%-2) ' remove CR/LF from byte
600 PRINT FETCH$                        ' print incoming strings
610 IF MID$(FETCH$,5,3) <> "END" THEN 540 ELSE RETURN
620 END
```

The following is a another interactive terminal program this time using the MetratByte DV-488 interface for the IBM-PC.

MetraByte

```
10 'This program uses the MetroByte DV-488 GPIB controller to move
20 'the Klinger DCS750 (axis 1) positioner to location +1000 and
30 'generate an SRQ at the end of the move.  Note this program requires
40 'DCS750 axis 1 dip switches to be set as follows:
50 'S1-1,2,3,4 (OFF), S1-5 (ON)
60 '-----------------------------------------------------------------
70 CLS
80 DATA "DH","SO 3","FO 146","PA+1000","END"
90 OPEN "$DV488" FOR OUTPUT AS #1
100 PRINT#1, "BUFFERCLEAR"
110 OPEN "$DV488" FOR INPUT AS #2
120 PRINT#1, "SYSCON MAD1=0 CIC1=1 BA1=&H300"  'GPIB controller address=0
130 PRINT#1, "STATUS 1" :INPUT#2, S%           'reset DCS750 SRQ status
140 INPUT "PRESS RETURN KEY TO START",Z$
150 READ A$ : IF A$="END" THEN 180
160 PRINT#1, "OUTPUT 1 $ !", A$ : GOTO 150 'send command w/CR terminator
170 '------------------------------------------------------------------
180 PRINT#1, "STATUS 1" : INPUT#2, S%      'serial poll DCS750
190 PRINT "Serial Poll =", S%
200 IF (S% AND 64) <> 64 THEN 180
210 PRINT "MOTION COMPLETE!"
220 '------------------------------------------------------------------
230 PRINT#1, "OUTPUT 1 $ !","TP"           'read position counter
240 PRINT#1, "ENTER 1 $ +"                 'terminate read on Line Feed
250 INPUT#2, POSITION$
260 PRINT POSITION$
270 END
```

The following is a Service Request (SRQ) sample program for one axis using the Hewlette Packard HP85 computer with an HPIB (IEEE-488) interface.

HP85

```
1 !    IMPORTANT:       1) Set axis #1 to address 1
2 !                     2) Previously issued "SO 3" (SRQ on motion complete)
3 !                    command assumed.
4 !
5 !
10 RESET 7          ! reset the HP85 HPIB interface
20 CLEAR 7          ! reset all devices on bus
30 CONTROL 7,16 ; 1 ! suppress line feed.  Enable carriage return.
40 ON INTR 7 GOSUB 140  ! go to line 700 if HPIB SRQ signal active
50 ENABLE INTR 7;8            ! enable interrupt
60 DATA "VA5000","AC20000","PA+1000","END"
70 DISP "PRESS RETURN KEY TO START"
80 INPUT Z$             ! dummy input
90 READ A$
100 IF A$ = "END" THEN 130    ! wait for interrupt
110 OUTPUT 701 USING "K"; A$  ! send command to device address 6
120 GOTO 90
130 GOTO 130                  ! wait for interrupt
140 S=SPOLL(701)        ! perform a serial poll
150 IF BIT(S,6)=1 THEN 200  ! go to line 200 if device # 1 caused SRQ
160 DISP "AXIS #1 DID NOT CAUSE SRQ"
170 RESET 7
180 ENABLE INTR 7;8
190 RETURN
200 CLEAR
210 DISP "** AXIS #1 REQUESTED SERVICE **"@DISP""
220 DISP "--THE STATUS OF AXIS #1 AT--"
230 DISP "--TIME OF INTERRUPT WAS--"@DISP""
240 IF BIT(S,0)=1 THEN DISP "+ LIMIT ACTIVE"
250 IF BIT(S,1)=1 THEN DISP "- LIMIT ACTIVE"
260 IF BIT(S,2)=1 THEN DISP "ORIGIN IN PROGRESS"
270 IF BIT(S,2)=0 THEN DISP "ORIGIN COMPLETE OR IDLE"
280 IF BIT(S,3)=1 THEN DISP "PROGRAM NOT COMPLETE"
290 IF BIT(S,3)=0 THEN DISP "PROGRAM COMPLETE"
300 IF BIT(S,4)=1 THEN DISP "MOTION NOT COMPLETE"
310 IF BIT(S,4)=0 THEN DISP "MOTION COMPLETE"
320 IF BIT(S,5)=1 THEN DISP "MOTOR ERROR"
330 IF BIT(S,5)=0 THEN DISP "NO MOTOR ERROR"
340 END
```

```
1 ' This program will create a text file which can be down-loaded to a
2 ' Klinger DCS750 controller, with firmware version 2.7 or later, for
3 ' sinusoidal motion operation.  After creating the text file this program
4 ' will call the down-loading program (SENDSINE.BAS) to transfer the file
5 ' via National Instruments GPIB controller (PCIIA) to the DCS750.  Note that
6 ' the text file should never exceed the DCS750 program buffer limit of 6K.
7 ' This can be controlled by using larger velocity update intervals (T)
8 ' i.e., 0.02 secs for larger amplitudes and shorter intervals (0.003 sec)
9 ' for smaller amplitudes at higher frequencies.  For best results suppress
10 'software limit operation and front panel position display refresh with
11 'DCS750 "FO" command.  The following DCS750 commands are especially
12 'important and should be well understood by the user/programmer:
13 '
14 '             EP, %, CP, QP, PA, AC, VS, DL, JL, FO, QS, ?
15 '
16 'Note that the "WT" command is not in the DCS750 User's Manual rev 2.05.04
17 'however, here is how it works:
18 '
19 '    COMMAND:  WT n
20 '      RANGE:  300 - 65000 (10uS increments)
21 '    PURPOSE:  This command will cause the DCS750 program to execute new
22 '            - "VS" instructions only at the specified WT period thereby
23 '              enabling precision time interval velocity updating.
24 '
25 SINE.BAS    Klinger Scientfic Corp
30 CLS
40 INPUT "     Target file (sine.dat)"; C$
50 INPUT "    Max amplitude (degrees)"; A
60 INPUT "           Frequency (Hertz)"; F
70 INPUT " Update interval (0.01 sec)"; T
80 PRINT:INPUT "Do you wish to change any parameters (N)"; YN$
90 IF YN$="Y" OR YN$="y" THEN 30
100 PI=3.141593
110 STAGE=(2000*5*36)/360              '(encoder)(gear box)(worm)/360 degrees
120 VMAX = INT((2*PI*F*A*STAGE)+.5)
130 ACCEL$=STR$(INT((2*PI*F*VMAX)+.5)) 'minimum acceleration required
140 IF T<.003 THEN T=.01               'minimum velocity update interval
150 IF C$="" THEN C$="sine.dat"        'default file name
160 N = INT(((1/F)*(1/T))+.5)          'where N is the number of update points
170 PRINT:PRINT "Vmax ="VMAX,,"Accel ="ACCELS,,"Points ="N
```

```
180 PRINT:INPUT "Hit RETURN/ENTER key to create file",DUMMY$
190 '------------------------------------------------
200 OPEN "o",#1,C$                    'open the target file
210 PRINT #1,"F0255"                  'suppress unnecessary options
220 PRINT #1,"KI300,KD4000,KI100,IL100,UF" 'RT300 filter parameters
230 PRINT #1,"EP"                     'DCS750 "open program buffer" command
240 PRINT #1,"OR1,WA1000,DH"          'initialize RT300 position
250 PRINT #1,"AC"ACCEL$               'required acceleration
260 PRINT #1,"WT"STR$(INT(T*100000!))  'DCS750 "WT" timer interval
270 PRINT #1,"DL A"                   'define "jump-to" target label
280 PRINT #1,"PA+"STR$(INT(2*A*STAGE)) 'peak-to-peak displacement
290 PRINT #1,"DL B"                   'define "jump-to" target label
300 '------------------------------------------------
310 FOR K=0 TO (N-1)/2                'creat half cycle profile
320 V=INT((VMAX*SIN((K*2*PI)/N))+.5)  'instantaneous velocity
330 VS$=STR$(INT((V*22.34778)+.5))    'convert to DCS750 "VS" command specs
340 VS$=RIGHT$(VS$,LEN(VS$)-1)        'remove ASCII space
350 PRINT#1,"VS"VS$                   'write velocity command to file
360 PRINT "*";                        'CRT prompt
370 NEXT K
380 '------------------------------------------------
390 PRINT #1,"PA0"                    'return to starting point
400 PRINT #1,"JL B 2"                 'repeat half cycle velocities twice
410 PRINT #1,"JL A"                   'repeat process indefinitely
420 PRINT #1,"%"                      'close DCS750 program
430 CLOSE :BEEP
440 RUN "sendsine.bas"                'download file to DCS750 via IEEE-488
```

```
1  'SENDSINE.BAS     Klinger Scientific Corp
5  '
10          CLEAR     ,59000!       ' Basic Declarations
20          IBINIT1 = 59000!
30          IBINIT2 = IBINIT1 + 3
40          BLOAD "bib.m",IBINIT1
50          CALL IBINIT1(IBFIND,IBTRG,IBCLR,IBPCT,IBSIC,IBLOC,IBPPC,IBBNA,
                              IBONL,IBSRC,IBSRE,IBRSV,IBPAD,IBSAD,IBIST,
                              IBDMA,IBEOS,IBTMO,IBEOT,IBRDF,IBWRTF)
60          CALL IBINIT2(IBGTS,IBCAC,IBWAIT,IBPOKE,IBWRT,IBWRTA,IBCMD,IBCMDA,
                              IBRD,IBRDA,IBSTOP,IBRPP,IBRSP,IBDIAG,IBXTRC,
                              IBRDI,IBWRTI,IBRDIA,IBWRTIA,IBSTA%,IBERR%,IBCNT%)
70 '-------------------------------------------------------------
80 ' This program is called by "SINE.BAS" to down-load and compile a
90 ' DCS750 program stored in a PC text file using National Instruments GPIB
100 '-------------------------------------------------------------
110 CR$=CHR$(13): QP$="qp"+CR$:CP$="cp"+CR$: EX$="ex"+CR$
120 '-------------------------------------------------------------
130 PRINT:INPUT "Enter down-load source file name (SINE.DAT):", SOURCE$
140 IF SOURCE$="" THEN SOURCE$="SINE.DAT"
150 CALL IBWRT(AXIS1%,QP$)      'force DCS750 into Command Mode
160 OPEN "I",#2,SOURCE$         'Open source file
170  WHILE NOT EOF(2)           'while file not empty...
180    INPUT#2,CMD$             'read next line in file and...
190    CALL IBWRT(AXIS1%,CMD$)  'transfer to DCS750
200  WEND
210 CLOSE                       'close source file
220 '-------------------------------------------------------------
230 CALL IBWRT(AXIS1%,CP$)      'compile DCS750 program
240 CALL IBWRT(AXIS1%,EX$)      'execute DCS750 program
250 '-------------------------------------------------------------
260 BEEP:PRINT :INPUT "Do you wish to do another transfer (Y) "; Y$
270 IF Y$="Y" OR Y$="y" OR Y$="" THEN 160 ELSE RUN "sine.bas"
280 END
```

RS232 Interface Software In BASIC

The following are sample programs using the RS232 communication link via IBM-PC BASIC language environment.

```
10 '*********************************************************************
20 'IMPORTANT: These programs assume the  RS232  input buffer was increased
30 '           to 5000 bytes with "/C" option in DOS  i.e.,"BASICA /C:5000"
40 '*********************************************************************
55 'This is an interactive RS232 terminal program for the DCS750
50 '
60 OPEN "COM1:9600,N,8,1,CS,DS,CD" AS #1 : CLS
70 PRINT: INPUT "ENTER DCS750 COMMAND:", CMD$: PRINT#1,CMD$
80 FOR DELAY=1 TO 500: NEXT     'Avoid premature exit on WHILE/WEND
90 WHILE LOC(1) <> 1: INPUT#1,LST$: PRINT LST$: WEND: GOTO 70
95 END
```

```
10 '
20 '
30 'This program will transfer (upload) commands stored in a DCS750 axis
40 'to a file in the PC.
50 '
60 OPEN "COM1:9600,N,8,1,CS,DS,CD" AS #1: CLS
70 INPUT "ENTER TARGET FILE NAME (DCS750.DAT):",TARGET$
80 IF TARGET$="" THEN TARGET$="DCS750.DAT"
90 OPEN "O",#2,TARGET$
100 INPUT "ENTER SOURCE AXIS (1):",AXIS: IF AXIS<1 or AXIS>30 THEN AXIS=1
110 CMD$=STR$(AXIS)+"LL": PRINT#1,CMD$: INPUT#1,LST$ 'get echo from buffer
120 FOR DELAY=1 TO 500:NEXT   'Avoid premature exit on WHILE/WEND
130 WHILE LOC(1)<>1: INPUT#1,LST$
140 IF MID$(LST$,10,2)="QP" OR MID$(LST$,5,3)="END" THEN 150  'QUIT
150 LST$=RIGHT$(LST$,(LEN(LST$)-9)): PRINT LST$: PRINT#2,LST$: WEND
160 BEEP: CLOSE
170 END
```

```
10 'This program transfers a file from the PC to the DCS750 program
20 'buffer
30 '
60 OPEN "COM1:9600,N,8,1,CS,DS,CD" AS #1: CLS
70 INPUT "ENTER SOURCE FILE NAME (DCS750.DAT):",SOURCE$
80 IF SOURCE$="" THEN SOURCE$="DCS750.DAT"
90 OPEN "I",#2,SOURCE$
100 INPUT "ENTER TARGET AXIS (1):",AXIS: IF AXIS<1 OR AXIS>30 THEN AXIS=1
110 PRINT#1,STR$(AXIS)+"EP"
120 WHILE NOT EOF(2): INPUT#2,LST$: PRINT LST$: PRINT#1,LST$: WEND
130 PRINT#1,"%": CLOSE: BEEP
140 END
```

CHAPTER 8
GENERAL PURPOSE I/O CONNECTOR

8.1 INTRODUCTION

The DCS750 provides two 37-pin General Purpose I/O (GPIO) connectors for all four axes. Axes 1 and 2 share GPIO "A" and axes 3 and 4 share GPIO "B". The pin assignment for these connectors is listed below.

GENERAL PURPOSE I/O CONNECTOR

1	+DIR JOG INPUT	(i)	*
2	-DIR JOG INPUT	(i)	*
3	HIGH SPEED INPUT	(i)	*
4	EMERGENCY STOP	(i)	*
5	NC	(i)	
6	ENCODER A (NOT)	(i)	
7	ENCODER B (NOT)	(i)	
8	ENCODER OUTPUT A	(i)	
9	ENCODER OUTPUT B	(i)	
10	ORIGIN SEARCH INPUT	(i)	*
11	INPUT BIT	(ii)	
12	SYSTEM RESET	(ii)	*
13	MOTOR POWER INHIBIT	(ii)	*
14	MOTOR DRIVE SIGNAL OUT	(i)	
15	POSITION RESET	(i)	*
16			
17	PROGRAMMABLE OUTPUT BIT	(i)	
18			
19	PROGRAMMABLE OUTPUT BIT	(ii)	
20	+DIR JOG INPUT	(ii)	*
21	-DIR JOG INPUT	(ii)	*
22	HIGH SPEED INPUT	(ii)	*
23	EMERGENCY STOP	(ii)	*
24	NC	(ii)	
25	ENCODER A (NOT)	(ii)	
26	ENCODER B (NOT)	(ii)	
27	ENCODER OUTPUT A	(ii)	
28	ENCODER OUTPUT B	(ii)	
29	ORIGIN SEARCH INPUT	(ii)	*
30	INPUT BIT	(i)	
31	MOTOR DRIVE SIGNAL OUT	(ii)	
32	POSITION RESET	(ii)	*
33	MOTOR POWER INHIBIT	(i)	*
34	+15 Volts (100mA MAX.)		
35	+5 Volts (100mA MAX.)		
36	LOGIC GROUND		
37	SYSTEM RESET	(i)	*

(i) Corresponds to axis 1 for GPIO "A" or axis 3 for GPIO "B".

(ii) Corresponds to axis 2 for GPIO "A" or axis 4 for GPIO "B".

* Indicates LOW level active.

All inputs are internally pulled-up to +5 Volts and all programmable output bits are open collector.

8.2 DETAILED DESCRIPTION OF PIN ASSIGNMENTS

PIN# DESCRIPTION

1 **+DIR JOG INPUT (i)**
When grounded, this input will cause the controller to jog in the "negative" direction at "OL" programmed low speed. See "DO" command.

2 **-DIR JOG INPUT (i)**
When grounded, this input will cause the controller to jog in the "positive" direction at "OL" programmed low speed. See "DO" command.

3 **HIGH SPEED INPUT (i)**
When grounded along with either jog input, this input will cause the controller to jog at "OH" command high speed. If grounded for 2-3 seconds without either jog input active and "FO" command bit-2 is reset, front panel program selection will be in effect. See section 3.7 Also see "DO" command.

4 **EMERGENCY STOP INPUT (i)**
Falling edge will cause the axis to abort motion and turn the motor off. The message "E STOP" appears momentarily on the front panel position display.

5 **NC**

6 **ENCODER A (NOT) OUTPUT (i)**
This is encoder signal A (not) as it appears on the motor cable

7 **ENCODER B (NOT) OUTPUT (i)**
This is encoder signal B (NOT) as it appears on the motor cable.

8 **ENCODER OUTPUT A (i)**
This is encoder signal A as it appears on the motor cable.

9 **ENCODER OUTPUT B (i)**
This is encoder signal B as it appears on the motor cable.

10 **ORIGIN SEARCH INPUT (i)**
Falling edge will cause controller to begin Origin Search execution. The axis uses the last "OR" parameter entered to determine if a floating or switch origin search should be performed. Also see "DO" command.

11 **INPUT BIT (ii)**
This input is tested by the "WH" and "WL" commands to generate conditional branches. It may also be monitored with the "QS", "TS", and "SO5" or "SO6" commands.

PIN# DESCRIPTION

12 **SYSTEM RESET INPUT (ii)**
A system reset is generated when grounded. All trajectory and PID parameters are restored and the motor is turned-off.

13 **MOTOR POWER INHIBIT INPUT (ii)**
When grounded, this input will disable all motor power. Note that a following error may occur as a result. See "FE" command.

14 **MOTOR DRIVE SIGNAL OUTPUT (i)**
This is the motor command signal output from the controller to the amplifier.

15 **POSITION RESET INPUT (i)**
Rising edge will cause the present location to be redefined as position zero. Equivalent to "DH" command. Also see "DO" command.

16 NOT USED

17 **PROGRAMMABLE OUTPUT BIT (i)**
The "SB" command sets this output "high". The "CB" command resets this output "low". This is an open collector output. This output may also be used to monitor motion active/idle status or origin search active/idle status. For more information see "DO" command.

18 NOT USED

19 **PROGRAMMABLE OUTPUT BIT (ii)**
The "SB" command sets this output "high". The "CB" command resets this output "low". This is an open collector output. This output may also be used to monitor motion active/idle status or origin search active/idle status. For more information see "DO" command.

20 **+DIR JOG INPUT (ii)**
When grounded, this input will cause the controller to jog in the "negative" direction at "OL" programmed low speed. See "DO" command.

21 **-DIR JOG INPUT (ii)**
When grounded, this input will cause the controller to jog in the "positive" direction at "OL" programmed low speed. See "DO" command.

22 **HIGH SPEED INPUT (ii)**
When grounded along with either jog input, this input will cause the controller to jog at "OH" command high speed. If grounded for 2-3 seconds without either jog input active and "FO" command bit-2 is reset, front panel program selection will be in effect. See section 3.7 Also see "DO" command.

PIN# DESCRIPTION

23 **EMERGENCY STOP INPUT (ii)**
Falling edge will cause the axis to abort motion and turn the motor off. The message "E STOP" appears momentarily on the front panel position display.

24 **NC**

25 **ENCODER A (NOT) OUTPUT (ii)**
This is encoder signal A (NOT) as it appears on the motor cable.

26 **ENCODER B (NOT) OUTPUT (ii)**
This is encoder signal B (NOT) as it appears on the motor cable.

27 **ENCODER OUTPUT A (ii)**
This is encoder signal A as it appears on the motor cable.

28 **ENCODER OUTPUT B (ii)**
This is encoder signal B as it appears on the motor cable.

29 **ORIGIN SEARCH INPUT (ii)**
Falling edge will cause controller to begin Origin Search execution. The axis uses the last "OR" parameter entered to determine if a floating or switch origin search should be performed. Also see "DO" command.

30 **INPUT BIT (i)**
This input is tested by the "WH" and "WL" commands to generate conditional branches. It may also be monitored with the "QS", "TS", and "S05" or "S06" commands.

31 **MOTOR DRIVE SIGNAL OUTPUT (ii)**
This is the motor command signal output from the controller to the amplifier.

32 **POSITION RESET INPUT (ii)**
Rising edge will cause the present location to be redefined as position zero. Equivalent to "DH" command. Also see "DO" command.

33 **MOTOR POWER INHIBIT (ii)**
When grounded, this input will disable all motor power. Note that a following error may occur as a result. See "FE" command.

PIN#	DESCRIPTION
34	+15 Volts (100mA max. from combined connectors A & B)
35	+5 Volts (100mA max. from combined connectors A & B)
36	Logical ground.
37	**SYSTEM RESET INPUT (ii)** A system reset is generated when grounded. All trajectory and PID parameters are restored and the motor is turned-off.

8.3 HANDHELD REMOTE

The DCS750 is available with a two(2) or four(4) axes remote option. In either case, they plug into one or both of the General Purpose I/O (GPIO) connecters provided in the rear panel.

All of the pushbutton switch functions available via the front panel position display option use the same inputs as does the GPIO. In fact, the DCS750 can not differentiate between GPIO or front panel switch closures.

Some GPIO inputs (i.e., JOG, HIGH SPEED, and POSITION RESET) also have useful double functions (see Chapter 3) like program selecting and jog speed programming. However, if your controller is not equipped with a display panel or your application incorporates a handdeld remote which plugs into the GPIO then it is highly recommended that you disable these double functions with "FO" command (bit-2).

A.1 REAR PANEL IEEE-488 CONNECTOR PIN ASSIGNMENT

1	D1	13	D5
2	D2	14	D6
3	D3	15	D7
4	D4	16	D8
5	EOI	17	REN
6	DAV	18	DAV GROUND
7	NRFD	19	NRFD GROUND
8	NDAC	20	NDAC GROUND
9	IFC	21	IFC GROUND
10	SRQ	22	SRQ GROUND
11	ATN	23	ATN GROUND
12	SHIELD	24	LOGIC GROUND

A.2 REAR PANEL RS-232-C CONNECTOR PIN ASSIGNMENT

1	NC	14	NC
2	RECEIVE DATA	15	NC
3	TRANSMIT DATA	16	NC
4	NC	17	NC
5	NC	18	NC
6	NC	19	NC
7	GROUND	20	NC
8	NC	21	NC
9	NC	22	NC
10	NC	23	NC
11	NC	24	NC
12	NC	25	NC
13	NC		

A.3 REAR PANEL MOTOR CONNECTOR PIN ASSIGNMENT

1	TACHOMETER +	14	GROUND
2	TACHOMETER +	15	TOP ZERO SIGNAL
3	TACHOMETER -	16	GROUND
4	TACHOMETER -	17	+ TRAVEL LIMIT
5	MOTOR + (10 AMPS MAX)	18	- TRAVEL LIMIT
6	MOTOR +	19	ENCODER CH. A
7	MOTOR -	20	ENCODER CH. B
8	MOTOR -	21	+5 OR +15 VOLTS
9	MOTOR + (1 AMP MAX)	22	GROUND
10	MOTOR +	23	ENCODER CH. A (NOT)
11	MOTOR -	24	ENCODER CH. B (NOT)
12	MOTOR -	25	NC
13	ORIGIN SWITCH SIGNAL		

A.4 ERROR MESSAGES (continued)

CODE	MESSAGE	NOTES
E13	(NEGATIVE HARDWARE LIMIT ACTIVE)	Negative travel limit switch located on positioner is active
E14	(POSITIVE HARDWARE LIMIT ACTIVE)	Positive travel limit switch located on positioner is active
E15	(NEGATIVE SOFTWARE LIMIT ACTIVE)	Programmable Negative travel limit is active ("SL-" command)
E16	(POSITIVE SOFTWARE LIMIT ACTIVE)	Programmable positive travel limit is active ("SL+" command)
E17	(EXCESSIVE FOLLOWING ERROR. MOTOR TURNED OFF)	Test for inverted motor phasing with "IP" command, or "VA" and/or "AC" target not possible
E18	(ENCODER QUADRATURE ERROR. MOTOR TURNED OFF)	Encoder error due to transient or poor signal quality. Position count not reliable now
E19	(NOT ALLOWED DURING MOTION)	Command can not be executed during motion
E20	(EXECUTABLE ONLY WITHIN PROGRAM)	"DL", "JL" and "JU" commands not allowed outside of program
E21	(MOTOR NOT CONNECTED)	Both, negative and positive hardware travel limits active
E22	(MEMORY CHECKSUM ERROR! SYSTEM RESET REQUIRED)	Memory error. Axis will purge memory with default parameters after next reset or power-cycle
E23	(PROGRAM COMPILED)	No syntax, parameter or jump to errors detected in the program
E24	(LABEL ALREADY DEFINED)	Duplicate labels not allowed. Only first letter recognized

A.4 ERROR MESSAGES

CODE	MESSAGE	NOTES
E00	(NO ERROR)	No errors detected
E01	(BAD COMMAND)	Command doesn't exist
E02	(ILLEGAL PARAMETER)	Parameter out of range or wrong parameter type
E03	(FATAL! SYSTEM RESET REQUIRED)	Hardware fault detected. Reset or power-cycle entire system
E04	(INSUFFICIENT MEMORY)	Program bigger than 4k memory
E05	(MISSING PROGRAM)	Program buffer empty
E06	(PROGRAM NOT COMPILED)	An attempt was made to execute a program that was not compiled
E07	(COMMAND BUFFER FULL)	512 byte input buffer was exceeded. Characters were lost
E08	(TARGET LABEL NOT IN PROGRAM)	Program line with "JL" or "JU" command missing target "DL"
E09	(NOT ALLOWED IN PROGRAM EXECUTION MODE)	Any commands affecting program structure (i.e., "DE 10-") are not permitted in PROG EXE mode
E10	(COMMUNICATION TIMEOUT)	DCS750 not able to establish or complete GPIB talk. It may be necessary to disable "END" line
E11	(RS232 FRAMING ERROR)	Verify host RS232 port set for 8 data, 1 stop, and no parity
E12	(RS232 OVERRUN ERROR)	Verify host RS232 port set for same baudrate as DCS750