DIGITAL AMPLIFIERS SOFTWARE MANUAL

Rev 3/93

How to use this manual

The DIGITAL AMPLIFIER SOFTWARE MANUAL complement the appropriate hardware manual that you received with your unit. It is divided into the following main four parts:

- **Communication**
- **Programming**
- **Fault Handling**
- **Instruction Set**

The **Communication** section describes the two communication buses that are available and especially the protocol of the RS485.
The **Programming** section provides the basic information of how to program the DCB.
The **Fault Handling** section describes how to utilize the various software routines that handle the protective features.
The **Instruction Set** section, describes each of the DCB instructions, restrictions on use and basic examples.

The last page of this manual is a short summary of all the commands - for the experienced user that just need to be reminded of all the various possibilities.

LIST OF CONTENTS

1. Communication.....4
1.1 RS232.....4
1.2 RS485 - Multidrop Serial Communication.....4
1.2.1 Elmo's RS-485 Protocol.....5
2. Programming.....7
2.1 Introduction.....7
2.2 Command Format.....7
2.3 Program Format.....8
2.4 Application Vs "Live" Programming.....9
2.5 Entering Programs.....10
2.5.1 Using the DCB Editor.....10
2.5.2 Executing Programs.....11
2.5.3 Debugging Programs.....12
2.5.4 Saving Programs.....12
3. Faults Handling.....13
3.1 Hardware Protection.....13
3.2 Programmable Error Limit.....13
3.3 Off-On-Error.....13
3.4 Automatic Error Routine.....14
3.5 Limit Switch.....14
3.6 Displaying the amplifier faults.....14
3.7 Storing the faults history.....15
4. Instruction Set.....16
4.1 Special Labels.....120

1. Communication

1.1 RS232

The RS232 in the DCB is configured as 8-bit, no parity, 1 stop bit, full duplex. The following baud rates are available: 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600. No hardware handshaking is required.

The only reliability test of the RS-232 communication bus is the echo check. By instructing the EO1 command (Echo ON) the DCB will echo each recognized character which is transmitted by the host computer.

1.2 RS485 - Multidrop Serial Communication

The RS485 in the DCB is configured as 8-bit, no parity, 1 stop bit, half duplex. The following baud rates are available: 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600. No hardware handshaking is required.

In the RS-485, which is a Half Duplex system, all the Transmitters and all the Receivers share the same Multidrop wire. Therefore, each character that is transmitted on the line, is automatically received by all the Receivers. This is an inherently "confused" way to transmit data and no "Echo" procedure can assure reliable communication.

In order to solve this reliability problem, it is necessary to use standard protocols procedures.

It is important to understand that using RS485 with the DCB products without any protocol is possible. This is also the defaulf condition whenever the RS485 is activated. However, the reliability of the communication is only assured when activating the protocol. This is done by sending the command CK1 from the host to the DCB.

The following chapter explains the standard protocol used and supplied by Elmo.

1.2.1 Elmo's RS-485 Protocol

Protocol structure

<- 8 bits ->	Remarks
%	Flag: Start Frame (ASCII)
%	Flag: Start Frame (ASCII)
Address	Address: DCB Number (BINARY)
Information	(ASCII)
"	"
"	"
"	"
"	"
"	"
CHKSUM	(BINARY)
;	End of CHKSUM protocol (ASCII)

Checksum Definition and Usage

The Checksum is the summation of all the protocol bytes transformed into two's complement form.

The result of summing the data bytes and the CHKSUM must be ZERO. If the result is not ZERO, there is an error in the received data.

The following characters are not allowed to be used as a CHKSUM number: 08,0D,3B,7E. They must be replaced as described in the following table:

Forbidden CHKSUM (HEX)	Replace to (HEX)
08	F8
0D	FD
3B	FB
7E	FE

EXAMPLE:

Checksum calculation for 2 data bytes.

```
     data 1:      1001000
     data 2:    + 0001011
                ------------
     sum          1010011
                ------------
One's complement: 0101100
                  +      1
                ------------
Two's complement: 0101101     (The CHKSUM byte in the protocol)
```

Checksum comparison

```
          sum :    1010011
                +
Two's complement:  0101101     (The CHKSUM byte in the protocol)
                ------------
                   0000000
```

Acknowledge and Negative Acknowledge

The acknowledgement procedure works as follows:

- The host sends data to the DCB by using the protocol.
- The DCB is performing the CHECKSUM test for the received data.
- If the result is zero, an ACKNOWLEDGE, ":", is sent to the host.
- If the results is different from zero, a NEGATIVE ACKNOWLEDGE, "?", is sent to the host.

Very Important:

The Checksum protocol cannot be used while programming or up/down loading.

2. Programming

2.1 Introduction

The DCB programming language is a powerful language that allows users to customize their motion application. In addition to standard motion functions, the language also provides several instructions for synchronizing motion with other process events. In this way, the DCB is a dedicated machine controller which can easily handle even the most sophisticated applications.

The DCB instruction set is BASIC-like and easy to use. Most instructions consist of two uppercase ASCII characters that correspond phonetically with the appropriate function. For example, the instruction BG begins motion, and ST stops motion.

The instruction set provides commands to specify parameters, initiate action, interrogate status and control program flow. For example, the GN command specifies the controller gain, AB aborts motion, TP reports the position and JP causes a program jump.

Program flow commands such as conditional jumps, event triggers, and subroutines allow the DCB to make its own decisions without a host computer. Almost any application can be programmed by combining program flow commands with motion commands.

To allow flexible programming, the DCB provides 128 user-definable variables. Variable are specified in a program and later assigned values. For example, the length in a cut-to-length operation may be specified as a variable which is later assigned by an operator input. Variable may be manipulated by arithmetic operations and functions.

The following sections describe the DCB instruction set and how to create programs.

2.2 Command Format

The DCB provides an extensive set of instructions for programming a variety of motion applications. Most instructions in the DCB instruction set are represented by two ASCII uppercase characters followed by applicable arguments. A semicolon or carriage return terminates the instruction.

Example:

PR4000;

PR is the 2 character instruction code (Position Relative). 4000 is the argument which represents the required position value. The ; terminates the instruction.

Whenever the DCB does not recognize a command, it returns a '?'. In this way, the host can verify that the information was correctly received by the DCB.

A complete listing of instructions is given in chapter 4: Instruction Set.

2.3 Program Format

A DCB program consists of several DCB instructions combined to solve a machine control application. Action instructions, such as starting and stopping motion, are combined with Program Flow instructions to form the complete program. Program Flow instructions evaluate real-time conditions, such as elapsed time or motion complete. and alter program flow accordingly.

Each DCB instruction in a program must be separated by a delimiter. Valid delimiters are the semicolon (;) or carriage return. The semicolon is used to separate multiple instructions on a single program line where the maximum number of instructions on a line is limited by 40 characters. A carriage return enters the final command on a program line.

All DCB programs must end with an End (EN) statement. It is convenient to start a new program with a Label but it is not a compulsory requirement. Labels start with the pound (#) sign followed by an uppercase letter.

Valid Labels

#A

#Z

There are also some special labels, which are used to define input interrupt subroutines, limit switch subroutines, error handling subroutines and automatic execution upon power-up subroutines.

Special Labels

#@ - Automatic Execution Program

#[- Limit Switch Routine

#] - Error Routine

#^ - Interrupt Routine

Example Program

```
#A              Beginning of the program
PR4000          Specify relative distance
BG              Begin motion
AM              Wait for motion complete
WT5000          Wait 5 seconds
JP#A            Jump to label A
EN              End of program
```

The above program moves the motor 4000 counts. After the motion is complete, the motor rests for 5 seconds. The cycle repeats indefinitely until the stop command is issued.

2.4 Application Vs "Live" Programming

The DCB provides two methods for command execution. One way is to send commands "live" over the communication port and have them executed immediately as the operator or computer program enters them. The other way is to execute an applications program residing in the DCB memory.

Executing application programs permit stand-alone operation without a host computer. Program features such as conditional statements and event triggers allow the controller to make its own decisions. For flexibility, however, variables allow program parameters to be specified by an operator. For example, the operator can enter the process speed, dwell times or part length.

Even while an application program is running, the operator can still send commands "live" via the communication line. For example, an operator can send the commands ST to Stop Motion or TP to interrogate the Position. This feature is often useful during initial machine set-up or program debugging. When both live communications and an

application program are running, the DCB will alternate command execution, so that neither command stream will be locked out.

2.5 Entering Programs

Application programs for the DCB may be created and edited either locally in the controller or remotely on another computer and downloaded to the controller.

2.5.1 Using the DCB Editor

The DCB provides a line Editor for entering and modifying programs. The Edit mode is entered with the ED instruction.

Note:
The ED command can only be given when the controller is in the non-edit mode, which is signified by a colon prompt.

In the Edit Mode, each program line is automatically numbered sequentially starting with 0. If no parameter follows the ED command, the editor prompter will default to the first line of the program in the memory. If desired. the user can edit a specific line number or label by specifying a line number or label following ED.

:ED	Puts Editor at the beginning of the program
:ED7	Puts Editor at line 7
:ED#A	Puts Editor at label #A

The DCB provides space for up to 50 program lines. Line numbers appear as 0,1,2 and so on. Program commands are entered following the line numbers. Multiple commands may be given in a single line as long as the total number of characters per line does not exceed 40. A semicolon separates each command on a line.

While in the Edit Mode, the programmer has access to special instructions for saving, inserting and deleting program lines. These special instructions are listed below:

Edit Mode Commands

<RETURN>

Typing the return key causes the current line of entered instructions to be saved. The editor will automatically advance to the next line. Thus, hitting a series of <RETURN> will cause the editor to advance a series of lines. Note, changes on a program line will not be saved unless a <RETURN> is given.

<cntrl>P

The <cntrl>P command moves the editor to the previous line.

<cntrl>I

The <cntrl>I command inserts a line above the current line. For example, if the editor is at line number 2 and <cntrl>I is applied, a new line will be inserted between lines 1 and 2. This new line will be labeled line 2. The old line number 2 is renumbered as line 3.

<cntrl>D

The <cntrl>D command deletes the line currently being edited. For example, if the editor is at line number 2 and <cntrl>D is applied, line 2 will be deleted. The previous line number 3 is now renumbered as line number 2.

<cntrl>Q

The <contrl>Q quits the editor mode. In response. the DCB will return a colon.

After the Edit session is over, the user may list the entered program using the LS command. If no operand follows the LS command, the entire program will be listed. The user can start listing at a specific line or label using the operand n. A command and new line number or label following the start listing operand specifies the location at which listing is to stop.

EXAMPLE:

```
:LS              List entire program
:LS7             Begin listing at line 5
:LS3,10          List lines 3 thru 10
:LS#A,11         List line label #A thru line 11
```

2.5.2 Executing Programs

A program can be executed using the XQ command. The XQ command can be followed by a label which defines the beginning of a program to be executed or by a number which defines the program line to start at. If no label is defined, the execution will start at the first program line. For example, XQ#A means execute the sequence starting at label #A.

Remember to quit the edit mode before executing a program.

2.5.3 Debugging Programs

The DCB provides trace and error code commands which are used in debugging programs. The trace command may be activated using the command, TR1. This command causes each line in a program to be sent out to the communications port immediately prior to execution. The TR1 command is useful for debugging programs. TR0 disables the trace function. The TR command may also be included as part of a program.

If there is a program error, the DCB will halt program execution at the line number at which an error occurs. This line will be displayed. The user can obtain information about the type of error condition that occurred by using the command, TC. This command reports back a number between 0 and 255, which correspond to an error condition. See chapter 4 - TC command - for details.

2.5.4 Saving Programs

The DCB provides SRAM memory for saving programs and parameters on power-down. Programs are automatically stored in the memory - no special user operation is required.

3. Faults Handling

3.1 Hardware Protection

The DCB includes hardware input and output protection lines for various fault conditions. These include:

Inhibit output	This signal goes low whenever the amplifier is inhibited.
Abort	A low input stops motion instantly without a controlled deceleration. Also aborts motion program.
Forward limit switch	Low input causes execution of limit switch subroutine, #[.
Reverse limit switch	Low input causes execution of limit switch subroutine, #[.

3.2 Programmable Error Limit

The DCB provides a programmable error limit. The error limit can set for any number between 1 and 32767 by using the ER(n) command. The units of the error limit are quadrature counts. The error is the difference between the commanded position and actual feedback position. If the absolute value of the error exceeds the value specified by ER, the DCB will shut the motor off if the OE1 instruction was previously given or will jump to the automatic error subroutine #], if it exists.

3.3 Off-On-Error

The software command, off-on-error (OE1) turns the motor off when the position error exceeds the limit set by the ER command. The program being executed is also aborted.

To re-enable the system, use the reset (RS), servo (SV) or servo here (SH) command. To disable off-on-error command OE0.

3.4 Automatic Error Routine

The #] label causes the statements following to be automatically executed if error exceeds the specified error limit. The error routine must be closed with the RE command. The RE command returns from the error subroutine to the main program. Note: The error subroutine will be entered again unless the error condition is gone. If OE in enabled, the error routine is inhibited.

3.5 Limit Switch

Activating the forward and/or reverse limit switches directs the controller to a special label for automatic execution of a limit switch subroutine. The #[label specifies the start of the limit switch subroutine. This label causes the statements following to be automatically executed if any limit switch is activated. The RL command ends the subroutine.

The state of the forward and reverse limit switches may also be tested during the Jump on Command statement. The condition LR specifies the reverse limit and LF specifies the forward limit.

3.6 Displaying the amplifier faults

Each amplifier's fault is stored immediately in the DCB RAM. In addition to that, a Failure Message is displayed. Following are all the valid Display Messages:

Event	Display	Display after Recurring
DIP switch 1 - ON	BAUD	OK
Load is under cont. current limit	CLIM	C-OK
Battery Low	BATT	B-OK
Abort condition (hardware only)	ABRT	A-OK
Amplifier's power stage disabled*	AMPD	H-OK
-15V out of limits	-15V	F-OK
Under or Over Voltage	VOLT	F-OK
+15V out of limits	+15V	F-OK
Over Temperature	TEMP	F-OK
Commutation problem (for brushless drives only)	CMMT	F-OK
Short condition at the power outputs	SHRT	F-OK

3.7 Storing the faults history

The DCB can store up to 10 fault events in a special register. Interrogating the faults history is done by the TF command. The information is transmitted on the communication line, line by line, when the last fault to occur is sent first. Clearing the faults register is done by the CE command.

The information received by the TF command does not mean that the last fault still exists - it may and it may not.

To check whether a fault still exist, the EA command must be used. By instructing EA, the controller sends a failure message on the communication line. If there is no failure when the command was sent, the DCB will respond with 0.

---

* The AMPD message appears in two cases:

1. When MO (Motor Off) command is given.
2. Position error exceeds the allowed value.

4. Instruction Set.

Each executable instruction is listed in the following section in alphabetical order. The instructions consisting of non-alphabetic characters such as %%, !! , and special labels are listed at the end. When you try to run the following sample programs, take care to place the command "E" in the end of your programs list.

A? – Tell Analog Input

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: In return to the A? command the DCB will report the value of the analog input. The answer will be either in Hex or Dec.
with a full range resolution of 11 bit.

EXAMPLES:

#A	Program A
JG4*A?	Define speed as four times the analog value
BG	Begin motion
WT10	Wait 10 ms
JP#A	Start again program A
E	

AB - Abort

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: Abort stops a motion instantly without a controlled deceleration. If there is a program operating, an abort command also aborts the program. For stopping a motion without stopping a program command AB1.

AC(n) – Acceleration

Restrictions:

While moving	NO
In a program	YES
Not in a program	YES
Default value	300,027

Description: This command sets the acceleration rate of the motor. The n parameter is an unsigned number in the range of 91 to 10^8. The parameter input will be rounded to the nearest factor of 91. The unit of the parameter is counts/s^2.

EXAMPLES:

AC1000000	Set the acceleration rate to 1000000 counts/s^2.
AC	Return acceleration.

AD(n) – After Distance

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The After Distance command is a trippoint used to control the timing of events. This command will hold up the execution of the following command until the position command has reached the specified relative distance (n) from the start of the move.

EXAMPLES:

#A	Program A
JG1000	Specify velocity mode and speed value
BG	Start motion
AD8000	After a distance of 8000 counts
SB1	Set bit no. 1.
EN	End program

AE – Amplifier Enable

Restrictions:

While moving	NO
In a program	YES
Not in a program	YES
Default value	---

Description: The AE instruction resets the amplifier after a protection latch. The protection latch would occur if LM1 was commanded.

AI(n) – After Input

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is a signed integer in the range of 1 to 7 decimal.

Description: The After Input command is used in motion programs to wait until after the specified input has occurred. If n is positive, it waits for the input to go to high. If n is negative, it waits for n to go low.

EXAMPLES:

#A	Program A
AI-1	Wait until input 1 is low
AC1000000	Acceleration is 1000000 counts/s^2
JG100000	Specify velocity mode and speed value
BG	Begin motion
AI1	Wait until input 1 is high
ST1	Stop motion
JP#A	Jump to program A
EN	End of program

AM – After Move

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The After Move command is a trippoint used to control the timing of events. This command will hold up execution of the following commands until the current move (motion profile) is completed. The completion of the move is the completion of the motion profiler - not the completion of the actual move of the motor.

EXAMPLES:

#A	Program A
PR15000	Position relative
BG	Begin motion
AM	After the move is complete
WT100	Wait 100ms
PA0	Position absolute 0 - origin
BG	Begin motion
AM	After the move is complete
JP#A	Jump to program A
EN	End of program

AP(n) – After Position

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is a signed integer in the range of $\pm 2^{30}$.

description: The After Position command is a trippoint used to control the timing of events. This command will hold up the execution of the following command until the absolute actual position of the motor (n) crosses the position specified.

EXAMPLES:

#A	Program A
DH	Define actual position as 0
JG1000	Jog mode
BG	Begin motion
AP4000	After position 4000
JG-1000	Change speed direction
AP-8000	After position -8000
CB2	Clear bit 2
ST	Stop
EN	End of program

AS - At Speed

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The At Speed command is a trippoint that occurs when the generated motion profile has reached the specified speed. This command will hold up execution of the following commands until the speed is reached. The AS command will operate after either accelerating or decelerating.

EXAMPLES:

#B	Program B
SP10000	Specify speed
PR25000	Specify position
BG	Begin motion
AS	After speed is reached
WT2000	Wait 2000ms
SP5000	Change speed
EN	End of program

BG - Begin

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The Begin command starts a motion.

EXAMPLES:

JG1000	Jog mode
BG	Begin motion

BS - Absolute R1 variable

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The BS command convert the R1 variable into an absolute value.

EXAMPLES:
In the following program, the BS command is used as to filter the noise on the analog input which is used as a speed reference.

```
SH
#L;JG0;BG
#A                  Program A
R1=A?;BS;JP#L,R1<24
JP#P,A?>0
JGA?+23*133;JP#A
#P;JGA?-23*133;JP#A
EN
```

CB(n) - Clear Bit

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is an integer in the range of 1 to 10

Description: The Clear Bit command clears one of the 10 bits on the output port. The Clear Bit (CB) and Set Bit (SB) instructions can be used to control the state of output lines.

EXAMPLES:

CB9	Clear output bit 9

CE – Clear the Fault's Buffer

Restrictions:	
While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The Clear Error command clears the fault's history buffer.

EXAMPLES:

CF(n) – Correction Factor

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is an integer in the range of $0\text{-}2^{30}$.

Description: The Correction Factor command is a feed forward constant used in conjunction with the Position Follower (PF) feature. The PF routine has a sampling rate of about 4ms. Each sampling period the controller receives an IP command and the follower speed is updated. When the Master motor has fast speed changes the above procedure yields in slow Slave response. The CF factor assists in improving the slave performance in such cases.
A low value will result in fast response of the follower while a high value for CF will result in a low response.

EXAMPLES:

CF10	Set the CF value to 10.
CF	The DCB will respond with the current value

CK(n) – CHECK SUM

Restrictions: For CK1 mode check sum protocol must be used on each end.
While moving: YES
In a program: YES
Not in a program: YES
Default value: CK0

Description: The CK1 command enables to use check sum protocol for communication transmitting receiving.
Each end of the communication network must be equipped with the check sum protocol.

CL(n.m) – Current Limit

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	Last value

Description: The Current Limit command sets the continuous current limit of the amplifier. The value of n.m should be higher than 5% of the value of MC.
Unit of n.m is in A.

EXAMPLES: Assume you need to run in a specified speed until you reach a mechanical stop and at that point to maintain a low pressure for 2 seconds.

```
00 #A               Program A
01 JG1000           Jog mode
02 BG               Begin motion
03 JP#B,PE>100      Jump to #B when the position error exceeds
                    100
04 JP3
05 #B               Program B
06 CL5              Set current limit to 5A
07 WT2000           Wait 2 seconds
08 ST1              Stop
09 EN               End of program
```

CP – Clear PROGRAM

Restrictions:

While moving	YES
In a program	NO
Not in a program	YES
Default value	---

Description: The Clear Program command clears all the user programs stored in the DCB RAM. The variables, and filter parameters are not cleared. The DCB will ask "ARE YOU SURE (Y/N)" before clearing the RAM.

DC – Decimal

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	DECIMAL

Description: The Decimal command sets the controller to interpret all future numbers as decimal number.

DH - Define Home

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The Define Home command defines the current position as the absolute zero.

EXAMPLES:

#A	Program A
JG-100	Jog mode
BG	Begin Motion
AI-3;ST1	Stop when input 3 goes low
WT100	Wait 100ms
DH	Define current position as zero.
DSP=00	Display P=00
EN	End of program

DL – Down Load

Restrictions:

While moving	YES
In a program	NO
Not in a program	YES
Default value	---

Description: The Down Load command transfers a data file from the host computer to the DCB. After entering the DL command the host has to send (in binary code) the number of lines to be transmitted. Instructions in the file will be accepted as a ASCII data stream without line numbers. The file is terminated using the \ command. Downloading a data file will clear any program in the DCB RAM. The data is entered beginning at line 0. If there are too many lines or too many characters per line, the DCB will return a ?.

EXAMPLES:

DL	Begin download
<cntrl>D	The binary code for four+1 lines of data
#A	ASCII data
PA170	ASCII data
BG	ASCII data
AM;SB1	ASCII data
EN	ASCII data
\	End download

DS(n) – Display

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The Display command displays either the characters that follows the Opcode or the information defined by these characters on the 4 digit display.

In some cases, it is expected that the data may exceed the 4-digit limitation. In such cases the user should extract the last 4 integer digits out of the desired variable by using the operation SS.

EXAMPLES:

DS,I?	Display the motor current.
DS,T?	Display the heatsink temperature.
DS"OK"	Display "OK"
V1=546799.1234	
SSV1; DS,SS	Display 6799

E – End of All Programs

Restrictions:

While moving	YES
In a program	YES
Not in a program	NO
Default value	---

Description: End of all programs. Following this instruction the controller will exit the editing mode. The E command must be written in a separate line.

EA(n) – Error Amplifier

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	last value

n is either 0 or 1

Description: The EA command returns the status of the amplifier's failures in one of the following selectable forms:

1) EA0 is initialy commanded.

For each (additional) EA command the controller will return a two-digit HEX number with the following meanings (when High):

Bit 7
Bit 6 Battery Low
Bit 5 -15V out of limits
Bit 4 Under/Over Voltage
Bit 3 +15V out of limits
Bit 2 Over Temperature
Bit 1 Commutation problem (for brushless drives only)
Bit 0 Short condition at the power outputs

2) EA1 initialy commanded.

The controller will automaticaly transmit a failure message on the communication line whenever an amplifier failure occurs. If the message was not received by the host it can be repeated by instructing EA. This time, the message will be transmitted without the F;. Following are the valid ASCII messages:

Message: F; -15V	Meaning: -15V out of limits
Message: F; Under or Over Voltage	Meaning: Under or Over Voltage
Message: F; +15V	Meaning: +15V out of limits
Message: F; Temperature	Meaning: Over Temperature
Message: F; Commutation	Meaning: Commutation problem (for brushless drives only)
Message: F; Short	Meaning: Short condition at the power outputs

ED(n) - Edit

Restrictions:

While moving	YES
In a program	NO
Not in a program	YES
Default value	---

n is either a label name or an unsigned integer in the range of 0 to 49 decimal.

Description: Enter the editing subsystem. In this subsystem,programs can be created, changed or deleted. The following commands are valid in this mode:
<CTRL>Q - Exit editing mode
<CTRL>P - Display the previous line of the program
<CTRL>I - Insert a line before the current line
<CTRL>D - Delete the current line<return>- Saves a line
DEL, INS, END, HOME * and right/left cursor arrows functions are valid inside a program line.

* The DEL, INS, END and HOME functions can be used with RS232 only.

EN – End

Restrictions:

While moving	YES
In a program	YES
Not in a program	NO
Default value	---

Description: The End command is used to designate the end of a program.

EXAMPLES:

#C	Program C
JG100000	Jog mode
BG	Begin motion
AI4	After input 4 is high
ST1	Stop
EN	End of program C

EO - Echo

Restrictions:	
While moving	YES
In a program	YES
Not in a program	YES
Default value	EO1 for RS232, EO0 for RS485

Description: The Echo command turns the echo on if EO1. If the echo is off(EO0), characters input over the communication bus will not be echoed back.

ER(n) – Error Limit

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	215

n is an unsigned integer in the range of 1 to 2^{15}

Description: The Error Limit command sets the magnitude of the position error, that will trigger an error condition.

EXAMPLES:

ER100	Set error limit to 100 counts
ER	What is the error range ?

FE – Find Edge

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The Find Edge command moves a motor until a transition is seen on the homing input. The direction of the motion depends on the initial state of the homing input (0 for forward, 1 for reverse). Once the transition is detected, the motor decelerate to a stop.

FR(n) - Following Ratio

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	Last value

n is a number in the range of ± 0.000001 to $\pm 2^{30}$.

Description: The Following Ratio command sets the ratio between the following motor motion to the auxiliary encoder (Master) motion.

EXAMPLES:

FR1	Set the following ratio to 1:1.
FR	The DCB will return the current value of FR.

FY(n) - Enable/Disable VY Command

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	FY0

n is either 0 or 1.

Description: The FY1 command enables the auxilary input velocity monito (see VY command) while FY0 disables it.

GN(n) - Gain

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	100

n is an integer in the range of 0 to 32767

Description: The Gain command sets the gain of the digital filter.

EXAMPLES:

GN150	Set the gain to 150
GN	What is the gain ?

HM(n) – Home

Restrictions:

While moving	NO
In a program	YES
Not in a program	YES
Default value	---

n is a signed integer in the range of 1 to 600000

Description: The Home command performs a two-stage homing sequence. The first stage consists of the motor moving at the user programmed speed until it sees a transition on the homing input. The direction for this first stage is determined by the initial state of the Homing Input (0 for forward, 1 for reverse). Once the homing input changes state, the motor decelerates to a stop.

The second stage consists of the motor changing direction and slowly approaching the transition again.

The third stage consists of the motor slowly moving forward until it detects transition to High at input 5 (from an index pulse or from any other source). It stops at this point and define it as position O.

n is the speed of motor slowly moving.

HX - Hex

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: Input/output in Hex

I? - Current?

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: Following the I? command the DCB reports the actual current in A.

EXAMPLES:

```
00 #A               Program A
01 V1=10000         Set variable value
02 JGV1             Define jog mode
03 BG               Start motion
04 JP#B,I?>5        Jump to program B if the current is
                    more than 5A
05 JP4             Jump to line 4 if the current is less than 5A
06 EN               End of program A
07 #B               Program B
08 V1=V1/10;JGV1    Reduce the speed by a factor of 10
09 WT10000          Wait 10 seconds
10 ST1              Stop
11 EN               End of program
```

II(n) - Input Interrupt

Restrictions:

While moving	YES
In a program	YES
Not in a program	NO
Default value	---

n is an integer in the range of 1 to 7.

Description: The Input interrupt enables the interrupt function for the specified input n. n equals 0 disables the Input Interrupts. If the specified input goes low during program execution, the program will jump to the subroutine with the label #^. Any trippoints set by the program will be cleared. The RI command is used to return from the #^ routine. The RI command also re-enables input interrupts.

IL(n) – Integration Limit

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	0

n is an integer in the range of 0 to 32767.

Description: The IL command sets a limit to the voltage that the integration part of the digital filter can reach.

EXAMPLES:

IL32767	Set the integration level to full signal amplitude
IL	What is the integration level ?

IN(xxx),Vn – Input Variable

Restrictions:

While moving	YES
In a program	YES
Not in a program	NO
Default value	---

xxx is a prompt message of length 33 characters or less
n is an integer in the range of 0 to 63 decimal.

Description: The IN command allows a variable to be input from a keyboard. When the IN command is executed in a program, The prompt message is displayed. The operator then enters the variable value followed by a carriage return. The entered value is assigned to the specified value. The IN command holds up execution of following commands in a program until a carriage return or semicolon is detected. If no value is given prior to a semicolon or carriage return, the previous variable value is kept. Input interrupts, Error interrupts and Limit Switch interrupts will still be active.
The IN command is valid also with the Rn variables.

EXAMPLES:
Operator specifies length of material to be cut in centimeters and speed in cm/sec. He needs a variable number of cycles of the same motion profile.

```
00 #A                          Program A
01 V5=1
02 IN Enter Distance(cm),V1    Prompt operator for length
03 IN Enter Speed(cm/s),V2     Prompt operator for speed
04 IN Number of Cycles,V6      Prompt operator for no. of cycles
05 V3=V1*500                   Convert units to counts
06 V4=V2*500                   Convert units to counts/sec
07 SPV4                        Speed command
08 PRV3                        Position command
```

09 BG	Begin motion
10 AM	After motion is complete

```
11 V5=V5+1                    Add 1 to the counter
12 JP6,V5<V6                  Re-start if the quantity is not
                              enough
13 EN                         End of program
```

IO – Inputs Variable

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is a positive integer in the range of 0 to 127

Description: The IO variable is an integer (either Decimal or Hex) that when translated into its binary value, represents the status of the 7 uncommitted inputs. Use the following table in order to interpret the IO variable:

Input no:	I7	I6	I5	I4	I3	I2	I1
Input binary value when high	1	1	1	1	1	1	1
Input decimal value when high	64	32	16	8	4	2	1

Masking a part of the inputs

It is possible to mask some of the inputs and to generate an IO variable that ignores the status of the masked inputs. This is done by the following instruction:

IO&(n)

n is a positive integer that when translated into a binary number, represents the inputs that are not masked and are used to generate the IO variable. For example: IO&19 means to load into the variable the value of inputs 1,2 and 5:

19 = 16(input 5) + 2(input2) + 1(input1).

Examples:

JP#A,IO&19=1 Jump to program A when input 1 is high and inputs 2 and 5 are low (no matter what is the status of inputs 3,4,6,7).

JS#P,IO&19=16 Jump to subroutine P when input 5 is high and inputs 1 and 2 are low (no matter what is the status of inputs 3,4,6,7).

IP(n) – Increment Position

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is a signed integer in the range of 1 to $\pm 2^{30}$.

Description: The Increment Position command permits commanding ne positions while the motor is in motion or at rest. It results in a profiled move according to the specified velocity and acceleration. No BG is required.

JG(n) - Jog

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	40000

n is a signed integer with no range restrictions.

Description: The JOG command sets the jog mode. The parameter following the JG, set the slew speed. It is recommended not to exceed a value of 600000 when using an optical encoder.

EXAMPLES:

JG10000	Set for jog mode with a speed of 10000counts/s
JG-2000	Set for jog mode with a speed of 2000 counts/s in the negative direction
JG	What is the speed ?

The JOG command may be executed during a motion if the controller is already in jog mode.

JP(n) – Jump

Restrictions:

While moving	YES
In a program	YES
Not in a program	NO
Default value	---

Description: The jump on condition command causes the program to jump to a line number (n), or to a program label, on the occurrance of the specified condition. The condition may be the status of inputs, outputs, motor or variable state. The use of logical operators and mathematical operation is permitted in condition statements. A jump without a condition is also permitted. After the JP command, the program must continue in a new line.

Valid Mathematical Operation

+	Add	!	Logical NOT (one operation only)
*	Multiply	&	Logical AND
/	Divide	|	Logical OR
-	Subtract		

Valid Logical Operators

>	Greater than
<	Less than
=	Equal to

Valid Condition Parameters:

Vn,Rn			A?	SP
PA			GN	ER
PR			KI	PX
In	(Input lines I1..I7)	ZR	PY	
On	(Output lines O1..O9)	AC	PE	
n	(numbers, including fractions)		JG	TZ
LF	(Forward limit switch)	I?		
LR	(Reverse limit switch)	T?	HM	
IO	(Reads 7 Inputs)	VY	TV	

OX (MOTOR OFF=1, MOTOR ON=0)

MX (MOTION COMPLETE=0 Else 1)

EXAMPLES:JP#C,V3=5 Jump to program C if V3 is equal to 5 JP#A,V1-V2>10 Jump to program A if V1 less V2 is greater than PE=2 Jump to line 7 if the position error is equal to 2

JP#B,I2=0 Jump to program B if input 2 is low

JP40,LF=0 Jump to line 40 if the forward limit switch is low

JS(n) - Jump to Subroutine

Restrictions:

While moving	YES
In a program	YES
Not in a program	NO
Default value	---

Description: The jump to subroutine on condition command will change the sequential order of execution of command in a program. If the jump is taken, program execution will continue at the line specified by the first parameter, which can be either a line number or label. The line number of the JS command is saved and after the next RT command is encountered, program execution will continue with the instruction JS. After the JS command, the program must continuein a new line.

Please refer to the JP command for a description of valid conditions.

JZ(n) – Zero Subroutine and Jump

Restrictions:

While moving	YES
In a program	YES
Not in a program	NO
Default value	---

Description: The JZ command is only valid in a program and is used to avoid returning from an interrupt (either input or error).
The JZ returns the stack to its original condition.
JZ adjusts the stack to eliminate one return. This turns the jump to subroutine into a jump.

It is used to allow returning from interrupt subroutine to any line number or to a program label.
When using the JZ(n) do not use:
-RI, RE, RL in the same subroutine.
-II6, and II7 (input interrupt I6, I7).

EXAMPLE:

```
0 #A;II1;DSLOOP;JP#A
1 #B;DS-JZ-;WT2000;JP#A
2 #^;MG INTERRUPT
3 SB1;WT1000;CB1
4 JZ#B
```

KI(n) - Integrator

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	1

n is an integer in the range of 0 to 32767

Description: The KI command sets the integral gain of the position loop's digital filter. The integrator term will reduce the position error at rest to zero.

EXAMPLES:

KI10	Specify integral gain as 10
KI	What is the value of KI ?

LM(n) – Latch Mode

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	Last value

n is either 0 or 1

Description: When instructing the Latch Mode command (LM0) the amplifier will start automatically after the cause of a protection disappears. The amplifier will be latched in Inhibit mode when commanding LM1. Clearing the inhibit should be done by AE.

LS(m,n) - List

Restrictions:

While moving	YES
In a program	NO
Not in a program	YES
Default value	0, Last line

m and n are either integers in the range of 0 to 49 or labels names

Description: The LS command sends a listing of the program in memory out the communication port. The listing will start with the line pointed to by the first parameter, which can be either a line number or a label. If no parameter is specified it will start with line 0.The listing will end with the line pointed to by the second parameter - again either a line number or a label. If no parameter is specified, the listing will go to the last line of the program.

EXAMPLE:

LS#B,25 List program B down to line number 25

MC(n) - Maximum Current

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	Last value

Description: The Maximum Current command informs the controller about the maximum rated current of the amplifier. The CL and PL commands will not be accurate if the MC is not correctly inserted.
n - current in A.

MG(n) - Message

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is any ASCII character or <cntrl>G

Description: The Message command sends up to 37 ASCII characters out the communication port. This can be used to alert an operator, send instructions, or return a variable value. The <cntrl>G or "Bell" character will sound a buzzer. If a new line is required put & before the beginning of the new line. A semicolon at the end of the message inhibits the carrige return line feed.

EXAMPLES:

MG V1=;V1=	Message: print V1 value
MG Error=;TE	Message: print position error

MO – Motor OFF

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: Motor off. Once the motion command is completed, the motor command is turned off and the amplifier is inhibited. This mode is useful when the motor shaft has to be turned manually. While the motor is turned, the position is still monitored by the controller.
The command SH will return the motor to the servo mode at the current position while the SV command will return the motor to servo mode at the position that the MO command was given.

NO – No Operation

Restrictions:

While moving	YES
In a program	YES
Not in a program	NO
Default value	---

Description: The No Operation command perform no action in a sequence, but can be used as a place holder. After the NO, up to 37 characters can be given to form a program comment. This helps to document a program.

OE(n) – OFF-On Error

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	1

n is either 0 or 1

Description: When n = 1, the motor command is turned off when the position error exceeds the error limit specified by the ER command. The motor switchs to the MO holding mode until a servo (SV) command is given. To avoid an abrupt transition of the motor, define the current position as zero (DH) before commanding SV. In order to reset this feature select n = 0.

OP(n) – Output Port

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	0

n is an integer in the range of 1 to 255 decimal

Description: The Output Port command sends 8 bits of data to the output port of the DCB. It is convenient to specify the HX mode before setting OP. You can use the output port to control external switches and relays. The n parameter is used to specify the number of bits effected starting with the LSB. The other bits are masked. For example, if n=3, only output bit 0 and 1 will be changed by OPn. If the n parameter is not specified, all bits will be changed.

EXAMPLES:

HX	Set Hex mode
OP 0	Clear output port
OP 85	Set outputs 8,3 and 1; clear the others

PA(n) – Position Absolute

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is a signed integer in the range of 0 to $\pm 2^{30}$.

Description: The Position Absolute command will set the final (absolute) destination of the next move.

PD(n.m) - Peak Duration

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	Last value

n is an integer in the range of 1 to 10 seconds m is an integer in the range of 0 to 9

Description: The Peak Duration command sets the duration of the maximum peak current.

PE – Position Error

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The PE command interrogates the position error of the controlled motor. It is used for loading the motor's position into variables or for any other programming purposes.

EXAMPLES:

V1=PE	Load the position error into V1
JP#B,PE>10	Jump to program B if the position error is higher than 10

PF(n) - Position Follower

Restrictions:

While moving	YES
In a program	YES
Not in a program	NO
Default value	0

n is an integer in the range of 0 to 2

Description: The PF1 and PF2 commands activate the position follower function and start the motion of the motor - no BG command is required. The DCB monitors the auxiliary encoder inputs and controls the motor position to follow the master encoder with the pre-specified following ratio. To stop the execution of the Position Follower (without stopping the execution of the program) use the PF0. Using the commands ST or AB will stop both the motion and the execution of the program. The best following performance will be achieved by using the PF2 command. Using this command the program execution will be stopped at the PF2 line. Using the PF1 command the program execution will not stop at the PF1 line. The correction factor (CF) must be optimized.

new parameters for PF command: V63,R63

V63 - The gain for position correction (0.001 till 10)
normal put V63=0.05

R63 - The gain for speed correction (1 till 50)
normal put R63=3.

Example:

```
0 #@;PF0
1 CF1000;V63=0.05;R63=3
2 FR1;TY2;FY1;WT100
3 PF2
4 EN
5 E
```

Notes:

1. Minimize the use of the communication bus during the execution of PF command. Use EO0 if you have to do it.
2. When executing PF1 do not use the following commands: WT,AI,AP,AD,AS,AM,FE,HM,IP.

PL(n.m) - Peak Limit

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	Last value

Description: The Peak Limit command sets the peak current limit of the amplifier. n.m - current in A.

PR(n) – Position Relative

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is a signed integer in the range of 1 to $\pm 2^{30}$.

Description: The Position Relative command sets the incremental distance and direction of the next move.

EXAMPLES:

#A	Program A
PR1000	Next move will be 1000 counts
BG	Begin motion
AM	After move
MG DONE	Message "DONE"
EN	End of program

PX - Position of the Main Encoder Input

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The PX command interrogates the position of the controlled motor. It is used for loading the motor's position into variables or for any other programming purposes. The returned number indicates the position in quadrature counts.

EXAMPLES:

V1=PX	Load the position into V1
JP#B,PX>1000	Jump to program B if the position is higher than 1000

PY(n) – Position of Auxilary Encoder Input

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is a signed integer in the range of $\pm 2^{30}$.

Description: The PY command sets or interrogates the value of the auxilary encoder counter. It can be used for dual loop or position follower applications. The returned number indicates the position in quadrature counts.

EXAMPLES:

```
#A              Dual loop program
PY0             Reset the auxilary counter
PR10000         Define the distance
BG              Start the motion
AM              After the move
WT10            Wait 10ms
JP#B,PY-PX>2    Go to the correction routine if the error is higher than 2 counts
EN              End of program A
#B              Correction routine
PRPY-PX         Define the correction distance
BG              Start the correction
EN              End of program B
```

RN m - Integer Variable

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is an integer in the range of 0 to 63

m is a integer number in the range of $\pm 2^{31}$.

Description: The Rn= command is used to define a variable which consists of an integer term only. Variables can be specified as integer numbers, constants, position or position error, or as the status of I/O lines. Once specified, a variable can be manipulated using arithmetic operation. Arithmetic operations proceed from left to right. The following commands' values can be loaded into the variables:

Vn,Rn		A?	SP
PA		GN	ER
PR		KI	PX
In	(Input lines I1..I7)ZR	PY	
On	(Output lines O1..O9)	AC	PE
n	(numbers, including fractions)	JG	TZ
LF	(Forward limit switch)	I?	
LR	(Reverse limit switch)	T?	HM
IO	(Reads 7 Inputs) VY	TV	
OX	(MOTOR OFF=1, MOTOR ON=0)		
MX	(MOTION COMPLETE=0 Else 1)		

EXAMPLES:

R2=4	Set R2 equal to four
R4=R1*R1	Set R4 equal to the square of R1
R3=PE	Set R3 equal to the position error
R5=I6	Set R5 equal to the status of input 6 (0 if low, 1 if high)
R1=	Return the value of R1
R1=R2-R3/3	Set R1 equal to one third of the difference between R2 and R3

RC(n) - Report When Complete

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	0

n is either 0 or 1.
O disable the function while 1 enables it

Description: The Report When Complete command will return an X to the host computer upon completion of the motion.

RE - Return from Error Routine

Restrictions:

While moving	YES
In a program	YES
Not in a program	NO
Default value	---

Description: End of an error subroutine

RI - Return from Interrupt Routine

Restrictions:

While moving	YES
In a program	YES
Not in a program	NO
Default value	---

Description: End of an interrupt routine

RL - Return from a Limit Switch Routine

Restrictions:

While moving	YES
In a program	YES
Not in a program	NO
Default value	---

Description: End of a limit switch rouitne

RS - Reset

Restrictions:

While moving	YES
In a program	NO
Not in a program	YES
Default value	---

Description: The Reset command resets the DCB parameters to its default values. If the @ label was used in a stored sequence then sequence execution will begin at that label.

RT – Return from Subroutine

Restrictions:	
While moving	YES
In a program	YES
Not in a program	NO
Default value	---

Description: End of a subroutine

SA – Set Address

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is an integer in the range of 1 to 127.

Description: The Set Address command sets the board address for RS485 communication. Note that the controller will stay addressed after an SA command until the next Wake Address (%%) command.

EXAMPLES:

SA1	Set controller address to 1
SA	Return the controller address

SB(n) - Set Bit

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is an integer in the range of 1 to 10

Dscription: Set output n to high.

SH – Servo Here

Restrictions:

While moving	No*
n a program	YES
Not in a program	YES
Default value	---

Description:

This instruction causes the controller to define the current motor position as the desired position and to switch to the servo mode (from the motor-off mode).

* Instructing the SH during the move will cause an immediate abort.

SP(n) – Speed

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	40000

n is an integer with no range restrictions.

Description: The Speed command sets the slew rate in n counts per second, with a resolution of 1. It is recommended not to exceed a value of 600000 when using an optical encoder.

EXAMPLES:

#A	Program A
PR10000	Define position mode and the travel distance
SP100	Set the speed
BG	Begin motion
EN	End of program

ST – Stop

Restrictions:	
While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: In response to the Stop command, the motor decelerates to a stop. If commanded during a program operation the program will be stopped also. The command ST1 will stop the motion without stopping the program.

SV – Servo

Restrictions:

While moving	NO
In a program	YES
Not in a program	YES
Default value	---

Description: The Servo command enables the position ontrol law and the amplifier. The motor will servo to its previously set command position.

T? - Tell Temperature

Restrictions:	
While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: Following the T? command the DCB reports the heatsink temperature in øc.

TC - Tell Code

Restrictions:

While moving	YES
In a program	NO
Not in a program	YES
Default value	---

Description: The Tell Code returns a number between 1 and 255. This coded number allows the user to determine why a command was not accepted.

Code	Meaning
1	Bad opcode
2	Command only from program
3	Command not valid in program
4	Operator error
5	Input buffer full
6	Number out of range
7	Command not valid while program is running
8	Command not valid while not running
9	Bad variable number
10	Empty program line or number
11	Invalid label or line number
12	Subroutine more than 8 deep
13	JG only valid when running in jog mode
14	ST or AB not valid while Motor Off
17	Edit not valid while program running
18	Return from a subroutine is missing
19	More than 40 characters in a line
20	Begin not valid with motor off
26	Label #^ not found
27	RI return error
34	LM629 not responding
35	CL > MC/2
36	PL > MC
37	The required command is being executed now.

38 Checksum error (for Elmo's checksum protocol only)

TD(n) – Time Derivative

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	1

n is an integer in the range of 1 to 256 in decimal.

Description: The TD command determines the sampling interval of the derivative term in the digital filter. Its unit is 409µsec.

EXAMPLE:

TD2	818µsec	(2*409µsec)

TE – Tell Error

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: In response to the Tell Error command the DCB will report the current position error.

TF – Tell Faults

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: In response to the Tell Faults command, the DCB will report the history of faults from the last one (present time) to the first one. The maximum number of faults that can be latched is 10.

History fault report:

"===RESET==="
"ABORT"
"CURRENT LIMIT"
"SHORT"
"COMMUTATION"
"+15V"
"-15V"
"UNDER/OVER VOLTAGE"
"AMPLIFIER DISABLE"

Example:

-After TF command you'll get the following message:

1: ===RESET===

Only power up reset was reported, this is the usual case when no failures occurred during the operation of the DCB.

-TF the message is:

1: "AMPLIFIER DISABLE"
2: "CURRENT LIMIT"
3: "===RESET==="

The amplifier was in current limit before it was disabled.
This leads to the conclusion that a position error
occurred because of the current limit.

TI - Tell Inputs

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: Tell inputs and status. In response, the system reports a three digit HEX number with the following binary translation: Bit 11 Forward limit switch

10 Reverse limit switch
9 Home input
8 Abort input
7 Always low
6 Input 7
5 Input 6
4 Input 5
3 Input 4
2 Input 3
1 Input 2
0 Input 1

TP – Tell Position

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The Tell Position command returns the absolute current position of the motor.

TR(n) - Trace

Restrictions:	
While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is either 0 or 1

Description: The trace command causes each instruction in a program to be sent out through the communication port prior to execution.TR1 enables this function and TR0 disables it. The trace command is useful in debugging programs.

TS - Tell Status

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The Tell Status command returns the status of the controller in the form of a two-digit HEX number.

Status if "High":

Bit 7	
Bit 6	
Bit 5	Abort
Bit 4	In motion
Bit 3	Motor OFF
Bit 2	Amplifier Error (see EA command for details)
Bit 1	Error limit exceeded
Bit 0	Program executing

TV – Tell Velocity

Restrictions:	
While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The Tell Velocity command returns the velocity in counts/s. Output is rounded to multiples of 2441.

TY(n) – Sampling Time of Speed Update

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is an integer which means msec. Only the following values can be used: 2,4,8,10,20,40,50,100,200,250,500.

Description: The TYn command is used in the position follower routine to determine the update rate of the follower speed. Typical values are 2-10msec.

TZ(n) - Tell Auxiliary Position

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is either 0 or 6 or 7.

Description: The TZ command latches the position of the auxiliary encoder input and loads it into the TZ register.

TZ - Tell the TZ register
TZ6 - Latch the position when input 6 goes low and load it to TZ
TZ7 - Latch the position when input 7 goes low and load it to TZ
TZ0 - Disarm the TZ6 or TZ7 commands.

UL - Upload

Restrictions:

While moving	YES
In a program	NO
Not in a program	YES
Default value	---

Description: The Upload command transfers ASCII data from the DCB to a host computer. After entering the UL command the DCB has to send (in binary code) the number of lines to be transmitted. Programs are sent without line numbers. Uploading is terminated with a <cntrl>Z.

EXAMPLES:

UL	Begin upload
<cntrl>D	The binary code for four+1 lines of data
#A	ASCII data
PA170	ASCII data
BG	ASCII data
AM;SB1	ASCII data
EN	ASCII data
<cntrl>Z	Terminator

Vn=(m) - Real Variable

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is an integer in the range of 0 to 63

m is a real number in the range of $\pm 2^{31}$.

Description: The Vn= command is used to define a variable. Variables can be specified as real numbers, constants, position or position error, or as the status of I/O lines. Once specified, a variable can be manipulated using arithmetic operation. Arithmetic operations proceed from left to right.

The following commands' values can be loaded into the variables:

Vn,Rn		A?	SP
PA		GN	ER
PR		KI	PX
In	(Input lines I1..I7)ZR	PY	
On	(Output lines O1..O9)	AC	PE
n	(numbers, including fractions)	JG	TZ
LF	(Forward limit switch)	I?	
LR	(Reverse limit switch)	T?	HM
IO	(Reads 7 Inputs) VY	TV	
OX	(MOTOR OFF=1, MOTOR ON=0)		
MX	(MOTION COMPLETE=0 Else 1)		

EXAMPLES:

V2=4	Set V2 equal to four
V4=V1*V1	Set V4 equal to the square of V1
V3=PE	Set V3 equal to the position error
V5=I6	Set V5 equal to the status of input 6 (0 if low, 1 if high)

V1=	Return the value of V1
V1=V2-V3/3	Set V1 equal to one third of the difference between V2 and V3

VR – Tell EPROM Version

Restrictions:	
While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The VR command returns the version of the EPROM that is installed on the DCB board.

VY – Tell Y Velocity

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

Description: The VY command returns the velocity of the auxilary encoder in counts/sec. The resolution is 1/TY counts/sec.
This command will work only after the FY1 command is initially instructed. The velocity information starts to be accurate only 100ms after the FY1 command.

EXAMPLE:

```
#A              Program A
FY1             Enable the VY command
WT100           Wait 100ms
#B              Program B
JGVY/2          Set Jog Mode with the speed value equals to
                half of the master speed.
BG              Start motion
#C              Program C
JGVY/2          Update velocity
WT10            Wait 10ms
JP#C
EN
```

This program is a velocity follower program. The follower speed is updated every 40ms.

WT(n) – Wait

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	---

n is an integer in the range of 0 to 65535ms

Description: The Wait command is a trippoint used to time events. After this command is executed, the controller will wait for the number of ms specified before executing the next command.

EXAMPLES:

The following program turns the DCB display to a "Digital Current Meter".

```
#A              Program A
DSI=,I?         Display the actual current
WT100           Wait 100ms
JP#A            Start from the beginning
 EN              End of progra
```

XQ(n) - Execute

Restrictions:

While moving	YES
In a program	NO
Not in a program	YES
Default value	---

n is a label or an unsigned integer in the range of 0 to 49.

Description: The Execute Program command starts a previously entered program operating. Execution will start at the label specified, or at the line specified, or at the first program in the memory if no label or line number are specified.

EXAMPLES:

XQ#A	Start execution of program A
XQ5	Start execution from line 5

ZM(n) – Zero the Auxiliary Position

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	0

n is either 0 or 6 or 7

Description: The ZM command sets the auxiliary position register to zero.
ZM6 - Set the auxiliary position to zero when input 6 goes low.
ZM7 - Set the auxiliary position to zero when input 7 goes low.
ZM0 - Disarm the ZM6 and/or ZM7 commands.

ZR(n) - Zero

Restrictions:

While moving	YES
In a program	YES
Not in a program	YES
Default value	500

n is an unsigned integer in the range of 0 to 32767

Description: The Zero command sets the derivative term in the control loop.

%%(n) – Wake Address All

Restrictions:

While moving	YES
In a program	YES (only through MG command)
Not in a program	YES
Default value	---

n is a number in the range of 0 to 127.

Description: The Wake Address (%%) command is used as part of the RS485 communication to enable a controller. All the DCBs will detect the % character, but only the currently active DCB will echo it. Whichever board's address matches the number following the %% sign will be the one to receive the following commands. There cannot be a space between the %% sign and the number.
The %%0 commad is used only when the Checksum is activated (CK1) and its meaning is Address All. Only controller no. 1 will echo this command.

EXAMPLE:

%%30	Enable board 30, disable all other

!! – Address All

Restrictions:

While moving	YES
In a program	YES (only through MG command)
Not in a program	YES
Default value	---

Description: The Address All command is a special command for RS485without the Checksum protocol. This command causes all controllers to accept the following command. There will be no echo.

EXAMPLES:

%%1	Address DCB 1
JG1000	Set joge mode
%%2	Address DCB2
PR10000	Set distance
!!;BG	Begin both axes

4.1 Special Labels

#@ – Automatic Execution Program

Description: The program which is specified by the label #@ starts automatically after every power on, reset input or RS command.

EXAMPLE:

```
#@                  Program for automatic execution
MO                  Motor off
MG GOOD MORNING     Send message "GOOD MORNING"
EN                  End of program
```

#[- Limit Switch Routine

Description: This label specifies the start of the limit switch subroutine. This label causes the statement follows to be automatically executed if any limit switch is activated. The RL command ends the subroutine.

Example:

```
#[                  Limit switch subroutine
AB1                 Abort motion (without aborting the
                    program)
MG LIMIT            Send message "LIMIT"
RL                  Return from #[ routine
```

#] – Error Routine

Description: The #] label specifies the start of the error limit subroutine. This label causes the following statements to be automatically executed if the error exceeds the value which is specified by ER command. The RE command ends the subroutine.

Example:

```
#]                  Error subroutine
AB1                 Abort motion (without aborting the program)
MG ERROR            Send message "ERROR"
RE                  Return from #] subroutine
```

#^ - Interrupt Routine

Description: The #^ label specifies the start of the Interrupt subroutine. This label causes the following statements to be automatically executed if the input specified by the II command goes low. The RI command ends the subroutine and also re-enables the II command.

Example:

```
#^                  Interrupt subroutine
V3=V3/2             Define new value for V3
RI                  Return from interrupt subroutine
```

~ - Clear Communication Buffer

Description: The ~ command clears the communication buffer (80 characters maximum). All the commands in the buffer are lost after this command is sent.

Example:

SUMMARY OF INSTRUCTIONS

Instructions may by grouped according to function: Motion, Control Settings, Program Flow, Programming-General, Error Handling & Status, Communication and I/O. These are listed below:

Amplifier Parameters

MC - Maximum Current
CL - Continuous Current Limit
PL - Peak Current Limit
PD - Peak Current Duration
LM - Latch Mode

Filter Parameters

GN - Gain
ZR - Zero
KI - Integration Constant
IL - Integration Limit
TD - Time Derivative

Motion Commands

AC - Acceleration Rate
JG - Jog
SP - Speed
PR - Position Relative
PA - Position Absolute
OE - OFF-ON-ERROR
ER - Error Limit
IP - Increment Position
FE - Find Edge
HM - Home

Start / Stop

BG - Begin
ST - Stop
MO - Motor OFF
SH - servo Here
SV - Servo
AB - Abort
RC - Report When Complete

Interrogate

TP - Tell Position
TE - Tell Error
TV - Tell Velocity
TI - Tell Inputs
TC - Tell Code
TS - Tell Status
I? - Tell Current
T? - Tell Temperature
A? - Tell Analog Input

General

OP - Output Port
TR - Trace
CB - Clear Bit
SB - Set Bit
DS - Display
DC - Decimal
HX - Hex
DH - Define Home
AE - Amplifier Enable
RS - Reset
EA - Error Amplifier
EE - Erase Error

Position Follower

PF - Position Follower
FR - Following Ratio
CF - Correction Factor

Programming

ED - Edit
AD - After Distance
AI - After Input
AM - After Move
AP - After Position
AS - At Speed
CP - Clear Program
DL - Down Load
E - End of All Programs
EN - End
II - Input Interrupt
IN - Input Variable
JP - Jump
JS - Jump to Subroutine
LS - List
MG - Message
NO - No Operation
PE - Position Error
PX - Position of the Main Encoder
Rn - Integer Variable
RE - Return from Error Routine
RI - Return from Interrupt Routine
RL - Return from a Limit Switch Routine
RT - Return from Subroutine
TR - Trace
UL - Upload
Vn - Real Variable
WT - Wait

XQ - Execute

Special programs Labels

#@ - Automatic Execution Routine
#[- Limit Switch Routine
#] - Error Routine
#^ - Interrupt Routine

Communication

EO - Echo
SA - Set Address
%% - Wake Address
!! - Address All

Auxiliary encoder (Y axis)

PY - Tell Y position
FY - Enable VY command
VY - Tell Y velocity