

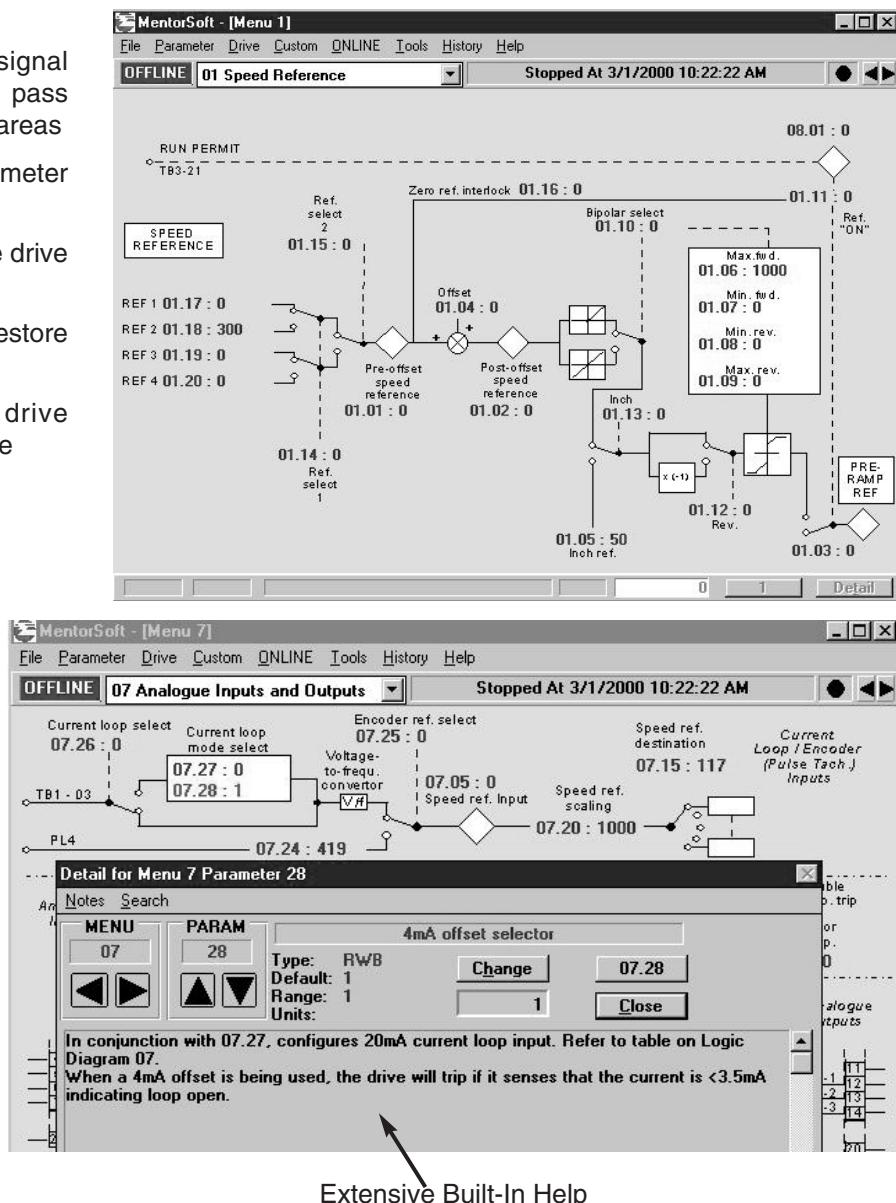
## 11.1 COMMUNICATIONS PACKAGES

There are a number of communication packages that can be used with the Quantum III to facilitate setup, record parameter data, view internal activity on a soft-scope and permit real-time interaction using soft meter, dial, sliders and other graphical animations such as bar graphs etc.

### 11.1.1 MentorSoft

Permits one to observe/trace signal flow as they come into the drive and pass through the various internal software areas

- Permits one to change any parameter via the PC
- Permits one to upload and save drive data to a file
- Permits one to download and restore a drive data file
- Permits one to compare the drive setup with a previously stored file

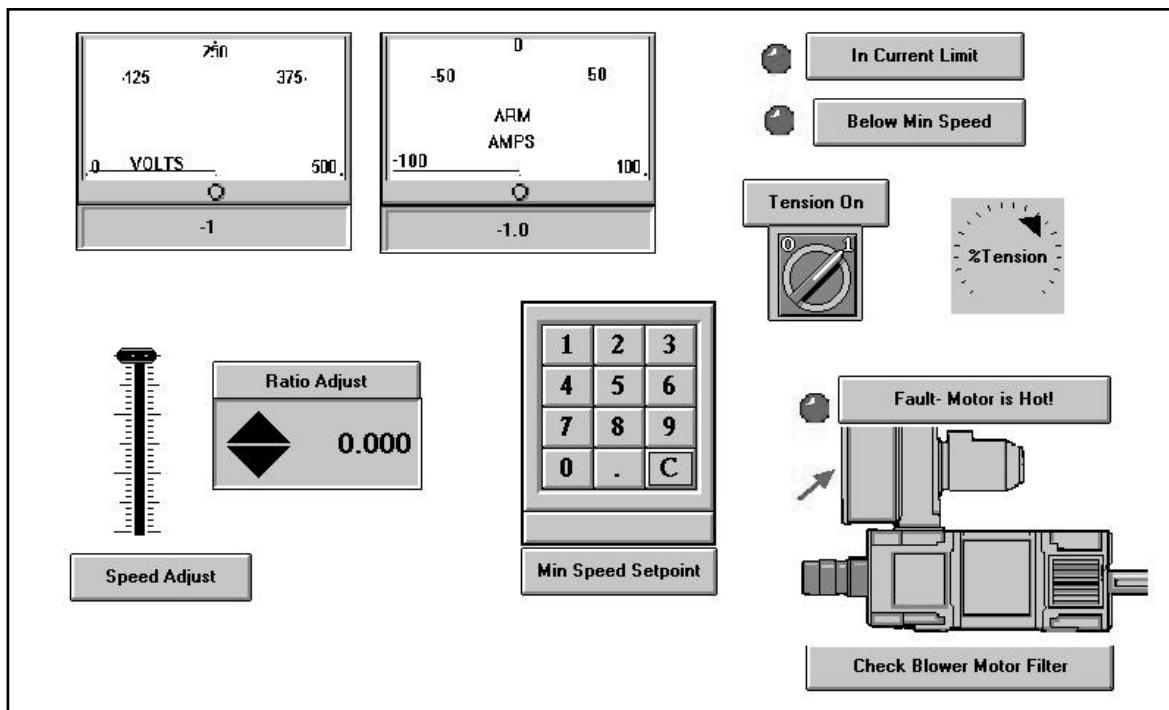


To obtain a copy of this program visit our website at:  
[www.ctdrives.com/downloads](http://www.ctdrives.com/downloads) under software.  
 Also see Appendix C Application Notes

# 11 Serial Communications

## 11.1.2 SystemWise

SystemWise is a SCADA-like software product that is excellent for setting up or tuning a drive. It permits you to observe internal drive data as a bar graph or analog style meter or as a scaled digital number expressed in your units. Conditions can be annunciated using soft LED's and you can control internal parameters using software Dials, Sliders and Increase/Decrease buttons. Graphics and custom photographs can be incorporated to permit you to customize your screens to your machine situation.



## 11.1.3 Factory Field Bus Communication Options

There are a number of popular communication options available for the Quantum III Drive listed in the table below.

Module	Description
MDIBS	Interbus S Communications Module-no coprocessor
MD24	Profibus DP Communication Module -no coprocessor
MD25	DeviceNet Communication Module-no coprocessor
9500-9100	Modbus Plus Communication Module-no coprocessor
MD29	Modbus RTU/Modbus ASCII Communications plus coprocessor

For additional information check out our website at  
[www.ctdrive.com/downloads](http://www.ctdrive.com/downloads)

A communications interface is standard in all Quantum III drives. It is a machine-machine interface, enabling one or more drives to be used in systems controlled by a host such as a process logic controller (PLC), computer, or Operator Interface (keypad).

Quantum III drives can be directly controlled and their operating configuration can be altered. Their status can be interrogated by such a host and continuously monitored by data logging equipment. A host can interface with up to thirty-two (32) Quantum III drives, Fig. 11-1, and up to 99 if line buffers are used.

The communication port of the drive unit is the connector PL2 (Fig.11-2). The standard connection is the RS485. Protocol is ANSI x 3.28 - 2.5 - A4, as standard for industrial interfaces.

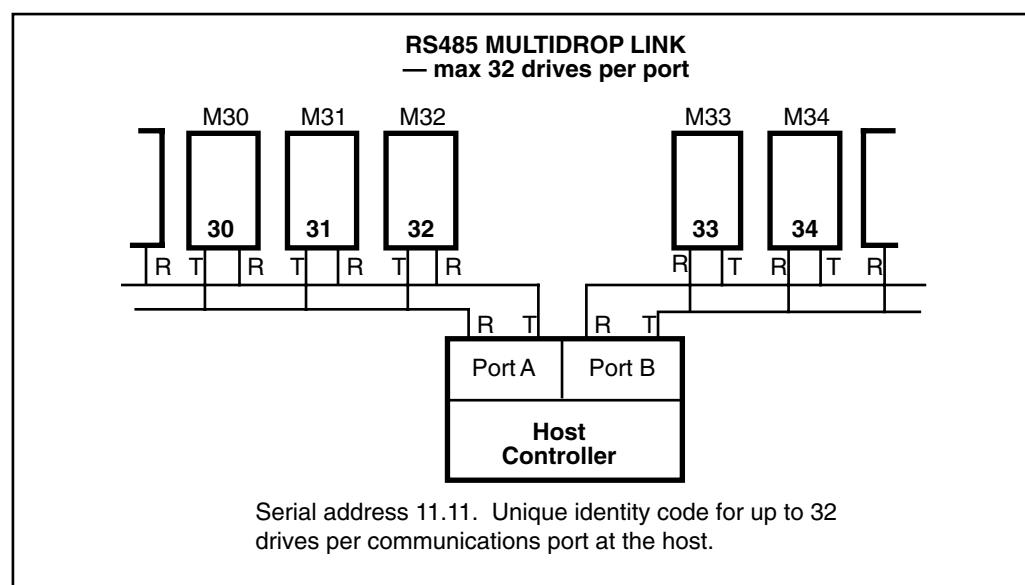
## 11.2 FUNDAMENTALS

Logic processors, such as computers, PLCs, and the communications systems of Control Techniques drives communicate by means of binary logic. Binary logic is ‘two state’, and is readily implemented by an electrical circuit which is either on or off. In Quantum III drives, the on-state is represented by a positive voltage, and the off-state by zero volts. The two voltages thus represent two distinct units of data, each being a binary digit (‘bit’) — either 0 or 1.

By fixing a time duration for each bit, a series of bits transmitted can be recognized by a receiver. If, also, a series or group always contains the same number of bits it becomes possible to construct a variety of different ‘characters’ that the receiver can recognize and decode. A group of four bits has sixteen (16) possible variants — 0000, 0001, 0010, and so on to 1111. Each of the sixteen variants represents one ‘hexadecimal’ character-unit — corresponding to the decimal numerals 0 to 9 followed by the six letters A to F — making 16 different and distinct characters.

The scope of the data that can be represented is much increased if two hexadecimal characters are combined to make a simple code. Since there are 16 hex characters, two in combination will produce  $16 \times 16 = 256$  possible different characters. Using this as the basis of a code, it becomes possible to represent a large number of symbols, or units of data, by means of only two hex characters, each of four bits, making eight bits in all and known as a ‘byte’.

Early in the development of computer technology it was recognized that a long stream of bits without, so to speak, any punctuation marks would be unmanageable and at risk of transmission errors passing unrecognized. The byte was adopted as a standard unit. To ensure that each byte is distinct, a start bit and a stop bit are added. The convention is that the start bit is a 0 and the stop bit a 1.



**Figure 11-1.**  
**Serial Address 11.11.**

# 11 Serial Communications

Each byte, therefore, occupies a finite time in transmission, but the interval between successive bytes is of no importance. Only the structure — the ‘framing’ or ‘character format’ — of the byte is significant. There is more than one convention for ‘framing’ the character. The format in Quantum III drives is ten bits as shown diagrammatically —

'Low' ASCII character byte									
1st hex character				2nd hex character					
Start bit	Seven data bits, variable							Parity bit	Stop bit
	0	lsb					msb		
The parity bit is used by the receiver of the message to check the integrity of the data byte.									

The character set used in Quantum III drives is the ‘low’ American Standard Code for Information Interchange (ASCII), comprising 128 characters, decimal numbered 0 to 127. The ‘Low’ ASCII Set is shown complete at the end of this Section. In the low ASCII set, the first hex character extends only from 0 to 7, binary 000, 001 etc to 111. A ‘start bit’, 0, is added to the beginning of the message, and a ‘parity bit’ and a ‘stop bit’, 1, are attached at the end.

The first 32 characters in the ASCII set (hex 00 to 1F) are used to represent special codes. These are the Control Codes, each of which has a particular meaning. For example, ‘start of text’ is STX, and, from a keyboard, is made by holding down the Control key and striking B once (Control-B). This is hex 02, and the actual transmission is the binary byte 0000 0010. The drive is programmed to know that this character signals that a command will follow. The control code at the end is EOT — ‘end of transmission’ — which tells all drives to look for a new message. If a host has a video screen, control characters appear on it in its format.

The components of all messages between the host and a Quantum III drive are formed of ASCII characters. The format of a message, i.e., the sequence in which the characters appear, is standardized for messages of each different kind, and is explained under Structure of Messages, in the next column.

## 11.3 PRELIMINARY ADJUSTMENTS TO THE DRIVE

Each drive requires a unique identity number, or serial address, set by parameter 11.11. The Baud rate 11.12 must be set to match the host. Data, drive status and the parameter set-up can be read from the drive in any mode, provided only that the drive is turned on, and that the serial address and Baud rate are correctly set.

## Communication Setup Parameters

When using the communication port, it is important that the PC comm port setting and the drive comm port setting match.

Param. #	Function	Range	Default Setting
#11.11	Serial Address of Drive	0 to 99	1
#11.12	Baud Rate	0 or 1 0=4800 1=9600	0 or 4800 baud
#11.13	Port Mode	1 to 4	1 = ANSI

## RS232 Connection

It is possible to communicate to the Quantum III directly from a Lap Top PC Compatible Computer using RS-232 communications, however it is not the recommended method. RS-232 communications is rather noise sensitive especially when used in industrial environments where drives are employed. Additionally, some PC's produce different voltage levels on their RS-232 outputs which can result in some PC's working ok and some not. At best, the cable length when using RS-232 would be as short as possible and never more than 10 feet.

Terminal designations for connector PL2 for RS422/485 communications interfaces is —

PIN NO.	RS232	RS485
1	NC	0V
2	TXD	TXD
3	RXD	RXD
4	—	—
5	—	—
6	0V	TXD
7	0V	RXD
8	—	—
9	—	—

## COMPUTER INTERFACE CABLE

### RS-232 to Quantum III

Computer DB-9 Female	Quantum III DB-9 Female PL2
3	3
2	2
8	4
7	5
5	1
Jumper 4-6	

Cable should be no more than 10' in length.

The serial port uses 7 data, 1 stop and even parity bits.

### Preferred Method

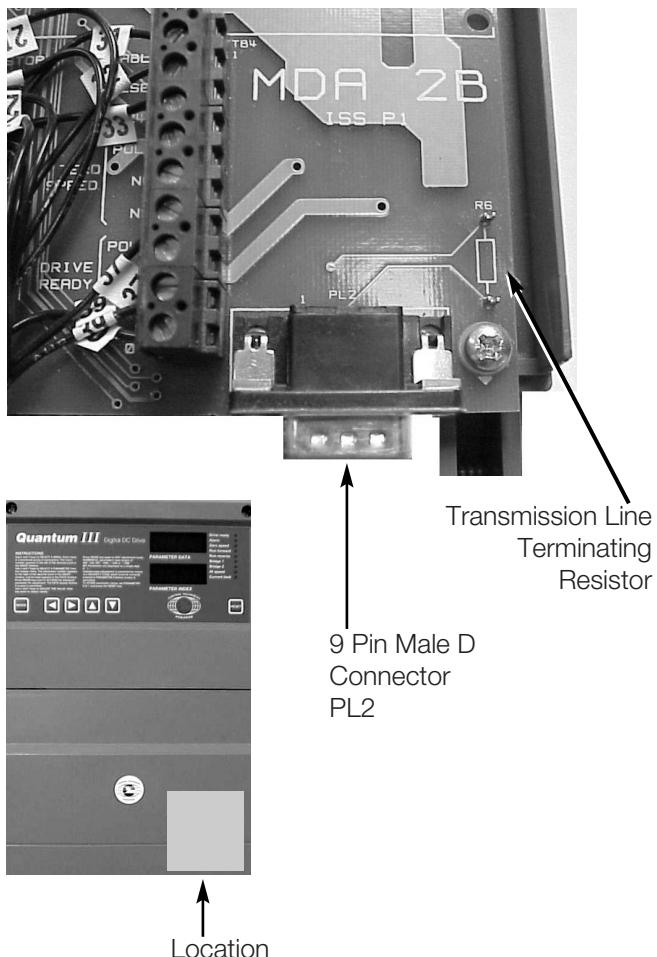
The recommended method of communication is using an RS-485 interface. From a PC, the use of an external RS-232 to RS-485 converter is recommended for temporary hookups. For a permanent communication situation such as when an Industrial PC is used as a SCADA (System Control And Data Acquisition) device, an RS-485 card placed within the PC would be the best option.

### RS-485 Port Wiring

To facilitate wiring one could elect to apply a Terminal Strip board as shown below. Part Number 9890-0056. It permits one to connect up using screw terminals rather than soldering a 9pin D female connector.



### RS-422/485 Communication Port



# 11 Serial Communications

## 11.4 RESOLUTION

Some parameters can be set to a higher resolution than that displayed or read by the serial interface. These are the real parameters with a range of  $\pm 1000$ .

If the user wishes to set variable to a higher resolution, six digits must be written in the data field. Quantum III then recognizes the request for higher resolution. For example, to set the speed demand to 47.65% of maximum speed, transmit —

+ 0 4 7 6 5

Refer to the following paragraphs for further explanation of the data field.

## 11.5 COMPONENTS OF MESSAGES

### 11.5.1 Control Characters

To conform to the standard structure of a message, the stages of a message are signalled by control characters. Each character has a specific meaning, a standard abbreviation, and is transmitted and received in ASCII code. If a message is initiated from a keyboard, the control characters are keyed by holding the Control key down while making a single-letter keystroke. Of the 32 control characters in the ASCII set, the seven in the table entitled "Control Characters in Quantum III Drives" are used in Quantum III serial communications.

### 11.5.2 Serial Address

Each drive is given an identity or address (parameter 11.11) so that only the drive that is concerned will respond. For security, the format is that each digit of the two-digit drive address is repeated, thus the address of drive number 23 is sent as four characters—

2      2      3      3

The serial address follows immediately after the first control character of the message.

### 11.5.3 Parameter Identification

For transmission by serial interface, parameters are identified by the four digits representing the menu and the parameter number, but without the decimal point, which is used in the text of this Manual for clarity. For example, to send 'menu 04, parameter 26', write 0 4 2 6.

### 11.5.4 Data Field

Data to be sent or requested occupies the next five characters after the parameter number. All of the operating parameters of the drive are **numerical** values, such as load, current, etc. The field for data is variable in length up to five characters maximum (but see reference to increased resolution in paragraph 11.4). No decimal point is used.

The state of **bit-parameters** is transmitted and received as real-value data, of value 0 or 1. Again, the format is flexible as long as no more than five characters are comprised, for example —

1  
0      1

— and so on.

CONTROL CHARACTERS IN QUANTUM III DRIVES			
CHARACTER	MEANING	ASCII CODE HEX	KEYED AS... CONTROL
EOT	Reset, or 'Now hear this' or End of Transmission	04	D
ENQ	Enquiry, interrogating the drive	05	E
STX	Start of text	02	B
ETX	End of text	03	C
ACK	Acknowledge (message accepted)	06	F
BS	Backspace (go to previous parameter)	08	H
NAK	Negative acknowledge (message not understood)	15	U

CONTROL	ADDRESS				CONTROL	PARAM				DATA				CONTROL	BCC
EOT	1	1	4	4	STX	*	1	1	7	-	4	7	6	ETX	<
CONTROL -D					CONTROL -B									CONTROL -C	

\* If this character happens to be a 0 as in this example, it can be written as a 0 or a space.

## 11.5.5 Block Checksum (BCC)

To permit the drive and the host to ensure that messages from one to the other have not become corrupted in transmission, all commands and data responses must be terminated by a block checksum character (BCC, paragraph 11.9).

## 11.6 STRUCTURE OF MESSAGES

### 11.6.1 Host to Drive

Messages from the host to the drive are of two kinds—

a request for information, or—  
a command.

Both kinds must start with the control character EOT (Control-D) to initiate the drive to receive a new message. This is followed by the serial address of the drive receiving the message. The format of the data and the choice of control character to terminate the message is different for the two kinds.

For an **information request**, sending the parameter number followed by ENQ instructs the particular drive addressed to supply data relating to that parameter.

For a **command**, a control character after the serial address tells the drive that the message is to be an instruction concerning its operational parameters, and that the next part of the message will be a parameter number and the instruction data. The instruction data occupies five to nine characters, or ten for high resolution. An instruction message is terminated by control character ETX followed by a block checksum (BCC, paragraph 11.9).

### 11.6.2 Drive to Host

Messages from the drive to the host are of two kinds—

a reply to a data request, or—  
acknowledgement of a message.

In **reply** to a data request, the start control character is STX, and is followed by the parameter number to confirm the request from the host, and then the five characters of data. The message is terminated by the control character ETX and a block checksum (BCC).

A message is **acknowledged** by the control character ACK if understood, or NAK if invalid, wrongly formatted or corrupt.

### 11.6.3 Multiple Drives

A message can be sent to two or more addresses simultaneously. If all drives are to respond to the same request or instruction, the message is transmitted to address 0 (zero).

## 11.7 SENDING DATA

Host command —

reset - address - start of text - menu and  
parameter - 1 to 5 data characters - end -  
BCC

For example, the message to the drive —

"change speed reference 1 of drive number  
14 to 47.6% in reverse"

would be sent as —

The drive will respond with an acknowledgement, either —

ACK if the message is understood and  
implemented, or —

NAK if the message is invalid, the data is too  
long, or the BCC is incorrect.

If a value sent is outside the limits for a parameter, the drive will respond with NAK.

# 11 Serial Communications

## 11.8 READING DATA

The drive will send any data to the host, provided that the request is valid. The format of a data request message is —

CONTROL	ADDRESS			PARAM			CONTROL
EOT	1	1	2	2	0	1	1
Control -D							ENQ

Host request —  
reset - address - parameter - end

For example, to find the speed set point 01.17 of drive number 12, send —

The drive replies in the following form —

start - parameter - 5 characters of data -  
end - BCC

For example —

CONTROL	PARAM			DATA			CONTROL	BCC
STX	0	1	1	7	-	0	4	7
Control -B							ETX	,

The reply first confirms that the data sent is the speed reference 1 (01.17); the five characters immediately following give the present setting as a percentage of full speed. The first character is either + or -, to indicate direction of rotation; the remainder is the numerical value. The message reads, “reverse at 47.6% of full speed” in this example.

### 11.8.1 Repeat Enquiry

The negative acknowledgement NAK (Control-U) can be used at a keyboard to cause the drive to send data repeatedly for the same parameter. It saves time when monitoring the value of a parameter over a period of time.

### 11.8.2 Next Parameter

To obtain data from the same drive for the next parameter in numerical order, send the positive acknowledgement ACK (Control-F). The drive will respond by transmitting the data relating to the next parameter in sequence.

### 11.8.3 Previous Parameter

To obtain data from the same drive for the previous parameter in numerical order, send backspace BS (Control-H).

### 11.8.4. Invalid Parameter Number

If the host sends a parameter number which the drive does not recognize, e.g. 1723, the drive will respond with EOT.

## 11.9 BLOCK CHECKSUM (BCC)

To ensure that data received can be verified, a block checksum is attached to the end of each command or data response. The BCC is automatically calculated by the sending logic and is derived in the following manner.

First, a binary exclusive-OR is performed on all characters of the message after the start-of-text command parameter.

For example, if the message to be sent to drive number 14 is —

“set speed reference 1 to 47.6% of full speed  
in reverse”

it is sent as —

Each of the separate digits,

0117 - (space or 0) 4 7 6 and Control-C

Reset	EOT (Control-D)
Serial address	1 1 4 4
Start of text	STX (Control-B) <i>Not included in BCC calculation</i>
Parameter	BCC calculation starts here
Reverse	0 1 1 7 (Menu no. and parameter no.)
476	- (a minus sign)
End of message	(space or 0) 4 7 6
	ETX (Control-C)
	finally, BCC, calculated as shown

is represented by a hexadecimal character and calculated in binary as shown in the following table. The XOR is shown progressively for each character.

CHARACTER	ASCII CHAR.	XOR
menu	0	011 0000
	1	011 0001 000 0001
parameter	1	011 0001 011 0000
	7	011 0111 000 0111
- (minus)	010	1101 010 1010
(space)	010	0000 000 1010
4	011	0100 011 1110
7	011	0111 000 1001
6	011	0110 011 1111
ETX	000	0011 <u>011</u> <u>1100</u>

The final XOR, underlined, is the BCC if its equivalent decimal value exceeds 31. As the ASCII characters from hex 00 to 1F are used only for control codes, the BCC has to exceed the value of 31 decimal. Whenever the XOR produces a (decimal equivalent) number less than 32, 32 is added. Thus, in the above XOR example,

011 1100 = 60 decimal, so that the BCC is character 60

for which the ASCII character is = <

Thus the complete message to set the speed of drive number 14, say, to 47.6% in reverse is as shown in the example message in paragraph 11.7.

