ABOUT THIS GUIDE

Change Summary – Revision D

August 1, 2002

This document, 88-019048-01**D**, supersedes 88-019048-01**C**. Changes and corrections are noted below.

Topic	Description
Node Address Configuration	• The software command `FBADDR0` is now required to configure the node address via hardware (default from the factory). Refer to page 4 for 6K PROFIBUS; and to page 18 for Gemini PROFIBUS.
`FBADDR` Command	• Range for the `FBADDR` command (Fieldbus Address) is now 0 – 125. The factory default value is now 0. Refer to page 8 for 6K PROFIBUS; and to page 28 for Gemini PROFIBUS.

Purpose of This Guide

This document is designed to help you implement the PROFIBUS features provided in your 6K and Gemini series products, as ordered with the PROFIBUS option. This publication addresses only the installation and programming tasks for the PROFIBUS features. For all other installation and programming instructions, refer to:

- *6K Series Hardware Installation Guide*, part number 88-017547-01
- *6K Series Command Reference*, part number 88-017136-01
- *6K Series Programmer's Guide*, part number 88-017137-01
- *Gemini GV6 Hardware Installation Guide*, part number 88-018364-01
- *Gemini GT6 Hardware Installation Guide*, part number 88-018374-01
- *Gemini Series Programmer's Reference*, part number 88-017778-01
- Refer also to the online help system in Motion Planner

What You Should Know

To install and troubleshoot your 6K and Gemini series products with the PROFIBUS option, you should have a fundamental understanding of:

- Electronics concepts, such as voltage, current and switches.
- Implementing and maintaining a given PROFIBUS network.
- Mechanical motion control concepts, such as inertia, torque, velocity, distance and force.
- Ethernet or serial (RS-232 or RS-485) communication, depending upon which communication protocol you are using.

⚠ WARNINGS ⚠

The 6K and Gemini products are used to control your system's electrical and mechanical components. Therefore, you should test your system for safety under all potential conditions. Failure to do so can result in damage to equipment and/or serious injury to personnel.

ALWAYS REMOVE POWER to the product before connecting any electrical devices (e.g., fieldbus connections, drives, encoders, I/O bricks, inputs, outputs, etc.).

Certification

6K and Gemini products with the PROFIBUS option have been PTO certified to comply with PROFIBUS-DP international standard EN 50170.

CHAPTER ONE

1 Implementing 6K PROFIBUS-DP

IN THIS CHAPTER

PROFIBUS Overview

6Kn-PB

The PROFIBUS option allows a 6K controller to be controlled via a PROFIBUS master, utilizing the PROFIBUS protocol for robust data exchange. The 6K is implemented as a generic PROFIBUS-DP device, allowing the user's application to fully define the data exchanged with a PROFIBUS master.

Cabling is not provided by Compumotor.

Technical Assistance

Technical questions regarding PROFIBUS should be addressed to your local PROFIBUS User Group. An address list is available on the PROFIBUS Internet site at www.profibus.com.

For support with 6K specific questions, contact Compumotor Applications Engineering at 800-358-9070, or e-mail us at tech_help@cmotor.com.

Implementation Process

PROFIBUS Master (user defined):

1. Use the provided CMTR090D.GSD file. Do not modify.
2. Configure communication baud rate.
3. Configure data packet size.

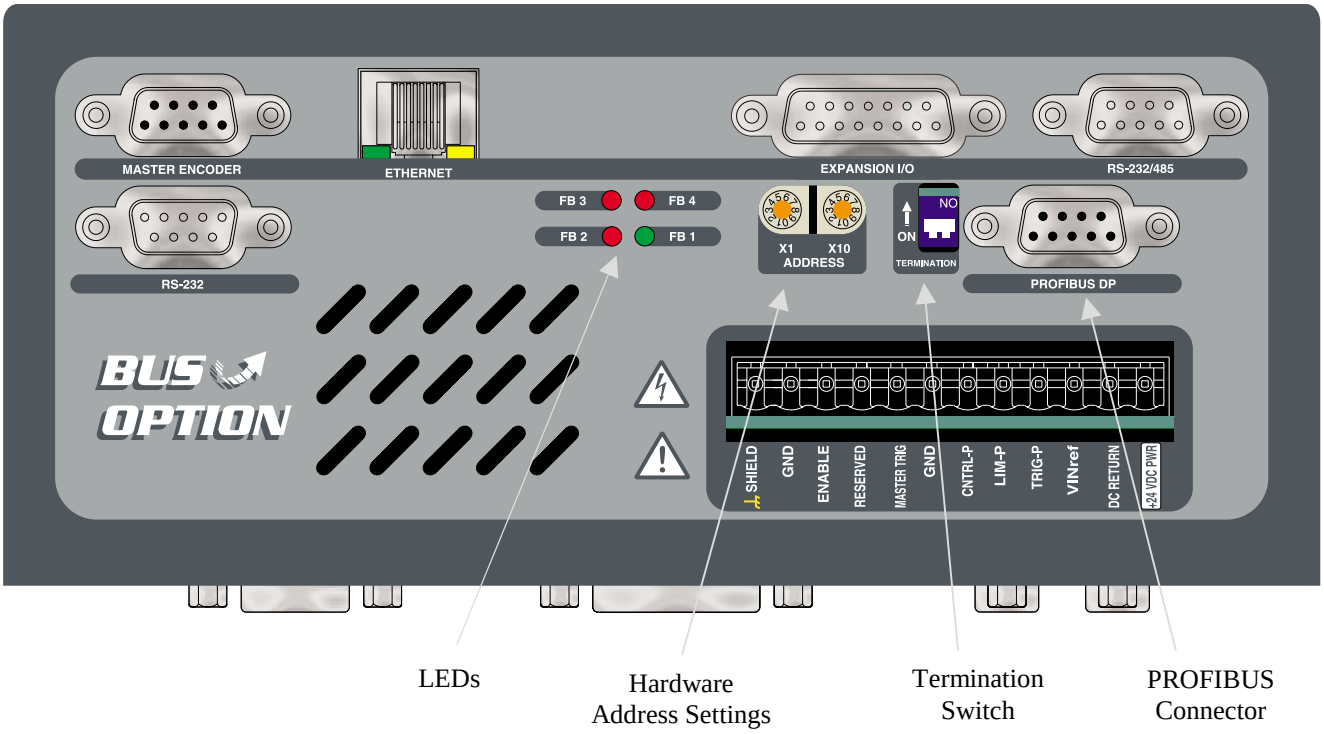
6K Controller:

1. Enable/disable terminating resistors as needed (see page 4).
2. Launch Motion Planner (CD-ROM is provided in your ship kit).
3. Establish a direct communication link (serial) with the 6K. Refer to the *6K Series Hardware Installation Guide* for connection instructions.
4. Configure node address (see `FBADDR` on page 8, or use hardware method on page 4).
5. Configure data packet size (`FBSIZE` must match master configuration). Refer to step 3 in the PROFIBUS Master implementation process above.
6. Reset the 6K controller to initialize the PROFIBUS card.
7. Write user code using `VARB1-VARB8` (depending on data packet size) for sending data from the 6K controller to the PROFIBUS master.
8. Write user code to read data from `VARB9-VARB16` (depending on data packet size) for receiving data from PROFIBUS master to 6K controller.

For more information, refer to the Programming Scenario on page 13.

GSD File

Each device in a PROFIBUS network is associated with a GSD file, containing all necessary information about the device. The latest version of the 6K GSD file (CMTR090D.GSD) can be downloaded from www.compumotor.com.



LED Status Indicators

Bicolor LED indicators are provided on the PROFIBUS option card. Refer to the following table for troubleshooting information provided by these LEDs.

LED	Steady	Flash	Function	Status *
FB1	--	--	Not used	--
FB2	Off		Module is not online	FBS bit #4= 0
	Green		Module is online	FBS bit #4= 1
FB3	Off		Module is not offline	FBS bit #4= 1
	Red		Module is offline	FBS bit #4 =0
FB4	Off		No diagnostics present	--
		Red	1 flash/second - FBSIZE setting does not match network configuration	--
		Red	4 flashes/second - Hardware failure	--

* To check status, execute the TFBS command (bit status report) or the TFBSF command (full text status report) in the terminal emulator. You can also use the FBS operator to assign or compare one or more status bits (e.g., use in an IF expression, assign to VARB variable, etc.). Refer to the TFBS command description on page 12.

PROFIBUS Connector Pin Out

The following table gives the pin out for the PROFIBUS Connector. The industry standard PROFIBUS connectors are used.

Pin	Name	Function
Housing	Shield	Protective earth
1	Not connected	-------
2	Not connected	-------
3	B-Line	Positive Rx/TxD
4	RTS	Request to send *
5	GND BUS	Isolated GND *
6	+5V BUS	Isolated +5V *
7	Not connected	-------
8	A-Line	Negative Rx/TxD
9	Not connected	-------

* +5V BUS and GND BUS are used for termination. Some devices like optical transceivers (RS485 to fiber optics) might require external power from these pins (not to exceed 80 mA). RTS is used in some equipment to determine the direction of transmission. In normal applications only A-Line, B-Line, and Shield are used.

Termination

If the 6K controller is used as the last node in a network, the termination switch must be in the ON position. Otherwise the switch must be in the OFF position. Please note, if an external termination connector is used, the switch must be in the OFF position.

Node Address

To configure the node address via hardware, two rotary switches are provided to set a node address of 1-99. The software command `FBADDR0` is also required to configure the node address via hardware (default from factory). Setting the rotary switches to 0 enables software configuration of node address (see `FBADDR` on page 8).

Example:

Switch x1 = 3 and x10 = 2, node address is 23

Baud Rate

The 6K will auto detect the baud rate of the PROFIBUS network. For a complete list of supported baud rates, see the CMTR090D.GSD file.

Data Format

When sending data over the fieldbus network, you can configure the data to be consistent over a word or the full data packet (see CMTR090D.GSD and FBSIZE on page 10). Most fieldbus masters will support full data packet consistency; however, some resource limited fieldbus masters may only be able to perform data consistency over a word.

The following table represents the internal binary variables and how they map to the data packet. Data is transmitted and received from left to right.

VARB1				VARB2			
MSW[1]		LSW[2]		MSW		LSW	
MSB[3]	LSB[4]	MSB	LSB	MSB	LSB	MSB	LSB

VARBn			
MSW		LSW	
MSB	LSB	MSB	LSB

[1] Most significant word, 16-bits, bits 31 to 16
[2] Least significant word, 16-bits, bits 15 to 0
[3] Most significant byte, 8-bits
[4] Least significant byte, 8-bits

Binary variables within the 6K programming language follow an unconventional format for bit assignment: bit 1 is the left-most bit and bit 32 is the right-most bit. When binary variables are exchanged with a PROFIBUS master, bit 1 may correspond to the right-most bit, and bit 32 may correspond to the left-most bit.

Example:

```
VARB1=h12345678      ; PROFIBUS Master receives, 0x87654321
VARI1=4PE            ; assume encoder position is +230
VARB1=VCVT(VARI1)    ; PROFIBUS Master receives, 0x000000E6
```

Implementing Data Exchange

It is up to the user's 6000 program to facilitate handshaking between the PROFIBUS master and the motion controller. There is no built-in handshaking or data synchronization performed by the motion controller (not to be confused with Sync mode).

To implement mailbox messaging (handshaking) between the 6K controller and the PROFIBUS master, you must set aside 2-bits/message within VARB1-16. One bit is used to acknowledge reading a message, a second bit is used notify the recipient a new message is available.

A message is user defined but could be used to control motion on a particular axis, update a task, update I/O, control a set of axes from a single message, or report motion status.

For example, if you would like to send a message from the PROFIBUS master to the 6K controller and then have the 6K controller generate a response message, the PROFIBUS master will use VARB9 bits 1 and 2, and the 6K controller will use VARB1 bits 1 and 2.

To send a mailbox message to the PROFIBUS master:

1. Make sure VARB1.1 is equal to VARB9.1; no unprocessed messages.
2. Place the message in VARB2 through VARB8.
3. Toggle VARB1.1 to indicate new message is available. VARB1.1 is now not equal to VARB9.1.

To receive a message from the PROFIBUS master:

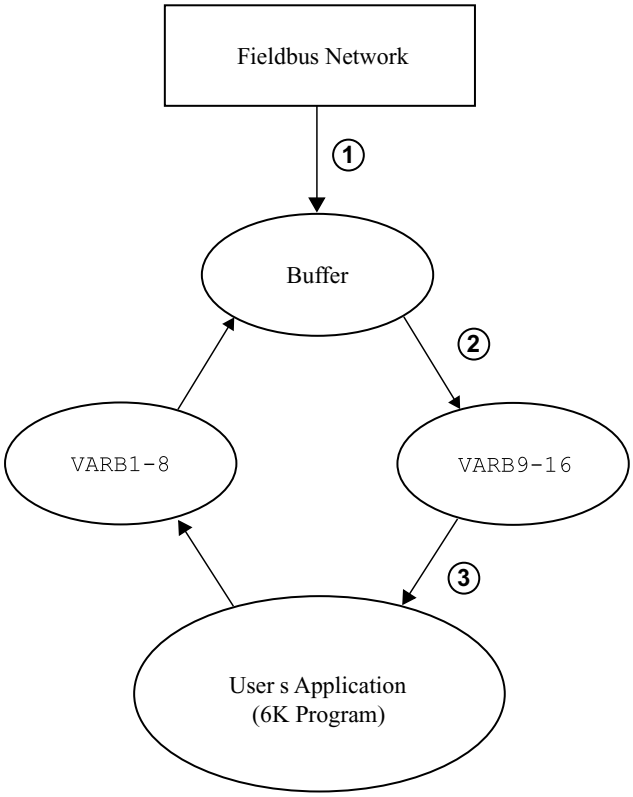
1. Make sure `VARB1.2` is not equal to `VARB9.2`, new message available.
2. Read the message from `VARB10-VARB16`.
3. Toggle `VARB1.2` to acknowledge reading the message. `VARB1.2` is now equal to `VARB9.2`.

The same operation would be repeated on the master side, except bits 1 and 2 would be reversed. An application scenario using mailbox messaging is provided on page 13.

Network Behavior

For complete list of supported features, see the CMTR090D.GSD file.

The PROFIBUS master has the option to send a Sync Control command to the 6K, which will then exchange `VARB9-16` only when a Sync command is received. The following diagram shows how data is exchanged in Sync mode.



1. 6K receives Sync Control command from PROFIBUS master.
2. 6K updates the contents of `VARB9-16` from the network.
3. User's application reads the new `VARB9-16` contents.

Note: `VARB1-8` will still be sent to the master regardless of Sync mode.

Handling a PROFIBUS Fault

If a PROFIBUS fault (Option card fault) occurs, the event causing the fault can be determined by checking the Fieldbus Status bit values (see FBS on page 9).

- If error bit #19 is disabled (ERROR.19-0), the controller performs a kill all when a PROFIBUS fault occurs.
- If error-checking bit #19 is set (ERROR.19-1), the controller performs a kill all and Error Status bit #19 is set (reported with ER, TER, and TERF). If an error program is assigned with the ERRORP command, the 6K controller branches (GOTO) to the program.

On power-up or out of reset, the events that can generate a fieldbus fault are ignored. This allows the PROFIBUS master time to commission individual nodes without causing the 6K controller to fault. Error bit #19 is edge sensitive to fault conditions.

To recover from an ER.19 fault, resolve the cause (see FBS on page 9) or reset the controller. To acknowledge the fault condition, issue the ERROR.19-0 command and then the ERROR.19-1 command. Refer also to page 13 for sample application scenarios.

Affected Commands and Features

When the PROFIBUS-DP option is installed, the following 6K commands and features are affected:

- Binary variables are affected by updates performed over the fieldbus network. See FBSIZE on page 10 for the exact binary variables affected by the fieldbus network.
- VARCLR will have no affect on binary variables assigned to the fieldbus network.
- A user's application will not be permitted to write to VARB9-16. If you attempt to change the state of VARB9-16, the controller will respond with an error message "VARB USED BY OPTION CARD" and the VARB command will not be executed; however, command processing will continue.
- A new bit definition for ERROR (bit #19) has been added for supporting the option card.
- A new function (option "I") has been added to the OUTFNC command to support detection of ERROR bit #19 being set.
- All commands preceded by "FB" or "TFB " (e.g., FBSIZE. TFBS, etc.) will be enabled when a fieldbus card is detected, and disabled when no fieldbus card is present or enabled.
- Ethernet will be disabled on the 6K when the PROFIBUS card is enabled (OPTEN1).

Command Descriptions

The following is a list of all the PROFIBUS specific commands. For a complete listing of 6K commands see the *6K Series Command Reference*.

ERROR Error Checking Enable

		Product	Rev
Type:	Communication Setup		
Syntax:	<!><%>ERROR<b><b>…<b><b> (32bits)	6Kn-PB	5.2
Units:	n/a		
Range:	b=0 (disable), 1 (enable), or X (don't change)		
Default:	0		
Response:	ERROR: *ERROR0000_0000_0000_0000_0000_0000_0000_0000		
See Also:	OUTFNC, FBS, TER, TERF		

A new bit assignment is added for the ERROR command, bit #19.

See FBS on page 9 for the event causing the error condition. To clear the error event, first resolve the cause, and then issue the ERROR.19-0 command followed by the ERROR.19-1 command. Error bit 19 is edge sensitive to error events.

In the event an option card fault occurs, VARB1-16 are cleared on the controller side.

Bit #	Function	Branch Type
19	Option card fault	GOTO

FBADDR Fieldbus Address

		Product	Rev
Type:	Communication Setup		
Syntax:	<!>FBADDR<i>	6Kn-PB	5.2
Units:	i = fieldbus address		
Range:	i = 0-125		
Default:	0		
Response:	FBADDR: *FBADDR3		
See Also:	FBSIZE, TOPSTS		

Use the FBADDR command to report the controller's current node address assignment and set the node address via software. The new value is saved into nonvolatile memory, and becomes effective after the controller is reset. Network configuration of node address is not supported. This command cannot report the hardware configuration setting. In order to set the node address via software, the hardware configuration method must be disabled (default from factory). Setting FBADDR0 is required to return to the hardware configuration method. See *Node Address* on page 4.

If the hardware configuration method is used to set the address, any attempt to set the address via software will be ignored and the message "CONFLICT WITH HARDWARE SETTING – DISABLE HARDWARE CONFIG FIRST" will be reported back.

Example:
Assume controller was assigned node address 1 out of reset:

```
>FBADDR
*FBADDR1
>FBADDR3        Set node address to 3
>FBADDR
*FBADDR3        New network setting will take effect after unit is reset!
```

[FBS] Fieldbus Status

Type:	Communication Setup	**Product**	**Rev**
Syntax:	See below	6Kn-PB	5.2
Units:	n/a		
Range:	n/a		
Default:	n/a		
Response:	n/a		
See Also:	ER.19, TFBS, TFBSF		

Use the FBS command to assign the fieldbus status to a binary variable or for use in a comparison command.

Example:

```
IF(FBS.4=b1)     ;Branch based on the status of FBS bit 4
```

The Fieldbus Status register bits are defined as follows:

Bit #	Function (1=Yes, 0=No)	Description
1	TIMEOUT [1,2]	Watchdog timed out. Controller has lost communication with fieldbus card
2	CHECKSUM FAULT [1,2]	Fieldbus card failed hardware check on boot-up
3	HWD CFG MODE	0 - Configuration set via software, 1 - Configuration set via hardware
4	ONLINE [1,3]	Controller is connected and data exchange is possible
5 -32	RESERVED	

[1] If any of these error conditions occur (bit #1 = 1, bit #2 = 1, or bit #4 = 0), the motion controller will perform a Kill (K command) on all axes. If error-checking bit #19 is enabled with the ERROR command (ERROR.19-1) the controller will also set error status bit #19 (ER, TER, and TERF) and branch to the ERRORP program.

[2] Error event is latched. Reset the controller to clear the error.

[3] Error event is recoverable, if error checking (ERROR) bit #19 is enabled and ERRORP program exists. If error bit #19 is disabled or no ERRORP program exists, the event becomes latched, and you will need to reset the controller to clear the error.

FBSIZE — Fieldbus Data Packet Size

		Product	Rev
Type:	Communication Setup		
Syntax:	<!>FBSIZE<i>	6Kn-PB	5.2
Units:	n/a		
Range:	i = 1-8		
Default:	8		
Response:	FBSIZE: *FBSIZE8		
See Also:	FBADDR, TOPSTS		

Use the FBSIZE command to set the number of binary variables exchanged with a PROFIBUS master. Data received or sent to the master is of the same size (cyclic), and each binary variable is 4 bytes. The new value is saved into nonvolatile memory, and becomes effective after the controller is reset.

Example:

```
FBSIZE8      ;Set fieldbus data packet size to 8 binary variables.
FBSIZE2      ;Set fieldbus data packet size to 2 binary variables.
```

Here are the variable assignments (from the controller's perspective) for each possibility of FBSIZE:

Command	Data Out	Data In
FBSIZE1	VARB1–VARB1	VARB9–VARB9
FBSIZE2	VARB1–VARB2	VARB9–VARB10
FBSIZE3	VARB1–VARB3	VARB9–VARB11
FBSIZE4	VARB1–VARB4	VARB9–VARB12
FBSIZE5	VARB1–VARB5	VARB9–VARB13
FBSIZE6	VARB1–VARB6	VARB9–VARB14
FBSIZE7	VARB1–VARB7	VARB9–VARB15
FBSIZE8	VARB1–VARB8	VARB9–VARB16

Regardless of FBSIZE setting, VARB1-16 are reserved for PROFIBUS activity and are not available for general use.

OPTEN — Option Card Enable/Disable

		Product	Rev
Type:	Communication Setup		
Syntax:	<!>OPTEN<i>	6Kn-PB	5.2
Units:	n/a		
Range:	r = 0 (disable) or 1 (enable)		
Default:	1		
Response:	OPTEN: *OPTEN1		
See Also:	TOPSTS		

Use the OPTEN command to enable (OPTEN1) or disable (OPTEN0) the option card on power-up. This feature allows Ethernet to be enabled when an option card is installed but disabled (if applicable). It also restores VARB1-16 for use by the user's application. Caution: If you later re-enable OPTEN1, VARB1-16 are then reserved for fieldbus activity.

NOTE: The new value is saved into non-volatile memory, and becomes effective ***after power is cycled***.

OUTFNC Output Function

		Product	Rev
Type:	Output		
Syntax:	<!><B>OUTFNC<i><-<a>c>	6Kn-PB	5.2
Units:	i = output #, a = axis, c = function identifier (letter)		
Range:	i = 1-32 (I/O brick dependent), a = 1-8 (depends on product), c = A-I		
Default:	c = A (programmable output function - default)		
Response:	OUTFNC: (function and status of onboard outputs) 1OUTFNC: (function and status of outputs on I/O brick 1) 1OUTFNC1: *1OUTFNC1-A PROGRAMMABLE OUTPUT - STATUS OFF		
See Also:			

An additional function (I) is added to the OUTFNC functions:

Identifier	Function Description
I	**Option Card Fault**: Output activates when error bit #19 is set for the option card fault. See the ERROR command for description of events. This requires ERROR.19-1 to be set, or the output will not activate. The OUTFNC-I command can only be assigned to task 0. If it is assigned to other than task 0, the error message "ALTERNATE TASK NOT ALLOWED" will be generated. OUTFNC-I cannot be assigned to a specific axis.

Example:

```
0%1OUTFNC8-i   ; Assign brick 1, output 8 to option card fault
0%OUTFNC1-i    ; Assign on-board output 1 to option card fault
2%OUTFNC1-i    ; Only task 0 allowed
2%ALTERNATE TASK NOT ALLOWED
```

TFBS Transfer Fieldbus Status

		Product	Rev
Type:	Communication Setup		
Syntax:	<!>TFBS<.i>	6Kn-PB	5.2
Units:	i = status bit number		
Range:	1-32		
Default:	n/a		
Response:	TFBS: TFBS.4: *1 (unit online or link ok, yes) *TFBS0001_0000_0000_0000_0000_0000_0000_0000		
See Also:	ER.19, [FBS], TFBSF		

The TFBS command provides information on the 32 fieldbus status bits. The TFBS command reports a binary bit report. If you would like to see a more descriptive text based report, use the TFBSF command.

Response for TFBS: *TFBS0001_0000_0000_0000_0000_0000_0000_0000

Bit#1…bit#32

For bit description, see FBS on page 9.

TFBSF Fieldbus Status Full Text

		Product	Rev
Type:	Communication Setup		
Syntax:	<!>TFBSF	6Kn-PB	5.2
Units:	n/a		
Range:	n/a		
Default:	n/a		
Response:	see example		
See Also:	ER.19, [FBS], TFBS, TOPSTS		

Use the TFBSF command to check the status of the fieldbus and display the status in full ASCII text to a terminal.

For status description, see FBS on page 9.

Example TFBSF **response**:

```
* TIMEOUT           NO        RESERVED     NO
* CHECKSUM FAULT    NO        RESERVED     NO
* HWD CFG MODE      NO        RESERVED     NO
* ONLINE            YES       RESERVED     NO
*
* RESERVED          NO        RESERVED     NO
* RESERVED          NO        RESERVED     NO
* RESERVED          NO        RESERVED     NO
* RESERVED          NO        RESERVED     NO
*
* RESERVED          NO        RESERVED     NO
* RESERVED          NO        RESERVED     NO
* RESERVED          NO        RESERVED     NO
* RESERVED          NO        RESERVED     NO
*
* RESERVED          NO        RESERVED     NO
* RESERVED          NO        RESERVED     NO
* RESERVED          NO        RESERVED     NO
* RESERVED          NO        RESERVED     NO
```

TOPSTS Option Card Status Full Text

		Product	Rev
Type:	Communication Setup		
Syntax:	<!>TOPSTS	6Kn-PB	5.2
Units:	n/a		
Range:	n/a		
Default:	n/a		
Response:	see example		
See Also:	OPTEN, TFBSF		

Use the TOPSTS command to check the status of the option card, and display the status in full ASCII text to a terminal.

Example TOPSTS **response:**

```
*6K OPTION CARD STATUS
*
*Option Card Enabled: Yes
*Option Card Type:   PROFIBUS-DP
*Option Card Firmware Rev:  92-018751-01-1.1
*Option Card Serial Number: 8-65535-65535
*
*6K PROFIBUS Product ID: 2317 (decimal)
*6K PROFIBUS Configured Via Software
*6K PROFIBUS Packet Size: FBSIZE8
*6K PROFIBUS Address: FBADDR1
```

Programming Scenario

NOTE: To understand the overall implementation process, refer to page 2.

```
;*******************************************************************************

DEL ERHND
DEF ERHND

;-------------------------------------------------------------------------------
;Fieldbus error event
;If the error event can be resolved, an unconditional jump is made to
;re-initialize the controller.
;-------------------------------------------------------------------------------
IF (ER.19 = b1)
       ;Insert application specific events to execute when a fieldbus error occurs.

       ;Wait for controller to go back online
       WAIT(FBS = b00X1)

       ; Controller back online
       ERROR.19-0          ;Acknowledge error event has been resolved
       ERROR.19-1          ;

       JUMP MAIN           ;Call to MAIN or other suitable initializer.
NIF

;-------------------------------------------------------------------------------
;Post power-up error event
;If the error event can be resolved, an unconditional jump is made to
;re-initialize the controller.
;-------------------------------------------------------------------------------
IF (FBS <> b00X1)
       ;Wait for controller to go back online
       WAIT(FBS = b00X1)

       JUMP MAIN
NIF

END ;ERHND program


;*******************************************************************************


;-------------------------------------------------------------------------------
;MAIN program
;
;In this program, the fieldbus error handler is assigned, enabled, and an output
;is activated when a fieldbus error occurs.
;
;Next a power-up check is made to determine if the 6k is active on the fieldbus.
;If not, the controller makes an unconditional jump to the error handler.
;
;After completing configuration and power-up checks, the controller begins
;exchanging data with the master. This section demonstrates mailbox messaging.
;
;-------------------------------------------------------------------------------
DEL MAIN
DEF MAIN

       ;Initialize controller
       ERRORP ERHND        ;Assign error handler program
       ERROR.19-1          ;Run ERRORP program (ERHND) when fieldbus error occurs
       OUTFNC8-I           ;Activate onboard output 8 if fieldbus fault occurs
```

```
        ;Post power-up check to verify no fieldbus errors exist.
        IF(FBS <> b00X1)      ;Check to see if it's online
                JUMP ERHND    ;Fieldbus error, jump to error program
        NIF


        ;Application's main loop
        L
                IF(VARB9.1 <> VARB1.1)
                        ;SEND NEW MESSAGE TO MASTER
                        WRITE"SENT NEW MESSAGE"

                        VARI1=4PE             ;Assign axis 4 encoder position to VARI1
                        VARB2=VCVT(VARI1)     ;Send encoder position out

                        VARB1=VARB1^H1        ;Notify master new message exists
                        T2
                NIF

                IF(VARB9.2 <> VARB1.2)
                        ;READ MESSAGE FROM MASTER
                        WRITE"GOT NEW MESSAGE"

                        VAR10=VCVT(VARB10)
                        A,(VAR10)             ;Assign new accel value

                        VAR11=VCVT(VARB11)
                        D,(VAR11)             ;Assign new distance value

                        VARB1=VARB1^H2        ;Acknowledge message received
                        T2
                NIF
        LN


END ; MAIN program

;*****************************************************************************

STARTP MAIN   ;Assign MAIN as the program to be run automatically on power-up and reset.
```