



1203-GK5 / 1336-GM5

1203-GU6 / 1336-GM6

Explicit Messaging using a 1756-DNB DeviceNet Scanner

APPLICATION NOTE

October 6, 1999

PURPOSE

The purpose of this document is to provide guidelines for wiring and control schemes for SCANport devices including Bulletin 1305 and 1336 PLUS AC Drives. This document is a suggestion only. Users must ensure that installations meet applicable codes and are suitable for the existing conditions.

INTENDED AUDIENCE

Personnel familiar with the hardware components and programming procedures necessary to operate DeviceNet and SCANport devices should use this application note. It is also assumed that the user has some familiarity with ControlLogix, the 1756-DNB scanner and ladder logic programming.

WHERE IT IS USED

The diagrams, parameter settings and auxiliary hardware used in this application note are designed to address specific issues in many different applications. Some changes by the user may be necessary to apply the concepts of this document to a specific application.

WHAT THIS NOTE CONTAINS

This document contains information and example ladder programs that demonstrate how to perform explicit messages using RSLogix5000 programming software, a Logix5550 controller, a 1203-GU6 or 1336-GM6 Enhanced DeviceNet to SCANport communications module, a 1756-DNB, and a SCANport product. The first program will execute a single parameter read. The second program will execute a scattered parameter read, and the third program executes a scattered parameter write. The information and example ladder programs can also be used with a 1203-GK5 or 1336-GM5 DeviceNet to SCANport communications module.

APPLICATION CONSIDERATIONS

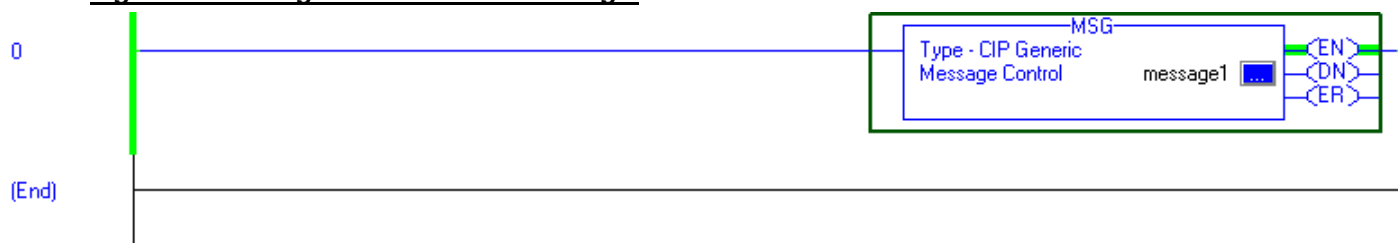
The examples were written to be simple and clear and do not fulfill all the functions needed for a real application. They contain no error handling. Consult the Logix5550, 1756-DNB, 1203-GU6 / 1336-GM6, and SCANport product manuals for more information. These examples assume a valid DeviceNet network already exists, a RSLogix5000 project has been created, and the 1756-DNB has been added to the I/O configuration.

Using explicit messaging to make frequent changes to a parameter will eventually result in the failure of the SCANport device's EEPROM (if so equipped). If an application requires frequent changes of only a few parameters, the parameters should be written using the SCANport Datalink function (if available) since this does not cause EEPROM writes to occur (not all SCANport products support Datalinks).

MESSAGE COMMAND CONFIGURATION

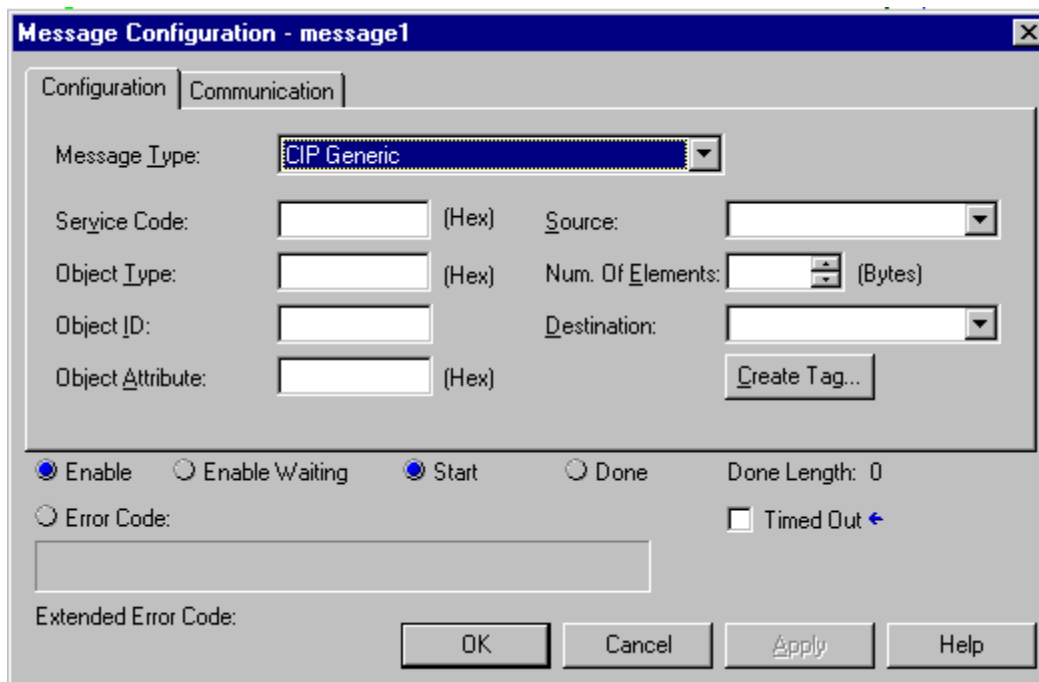
To send an explicit message over DeviceNet, Logix5550 uses message commands. The program below explains the configuration for the message instruction. Add a message instruction and create a message control tag ('message1' in this example).

Figure 1 –Message Command Ladder Logic



Click on the blue box inside the message command and a window like the one below pops up.

Figure 2 –Message Command Configuration



'Message Type' - must be CIP Generic.

'Service Code' - specifies type of service to be performed such as a get or send. Value entered in Hex.

'Object Type' - defined as Object Class in 1203-GU6 User Manual. Value entered in Hex.

'Object ID' – defined as Instance in 1203-GU6 User Manual. Value entered in Decimal.

'Object Attribute' – defined as Attribute in 1203-GU6 User Manual. Value entered in Hex.

'Source' – tag for any additional service data to be sent.

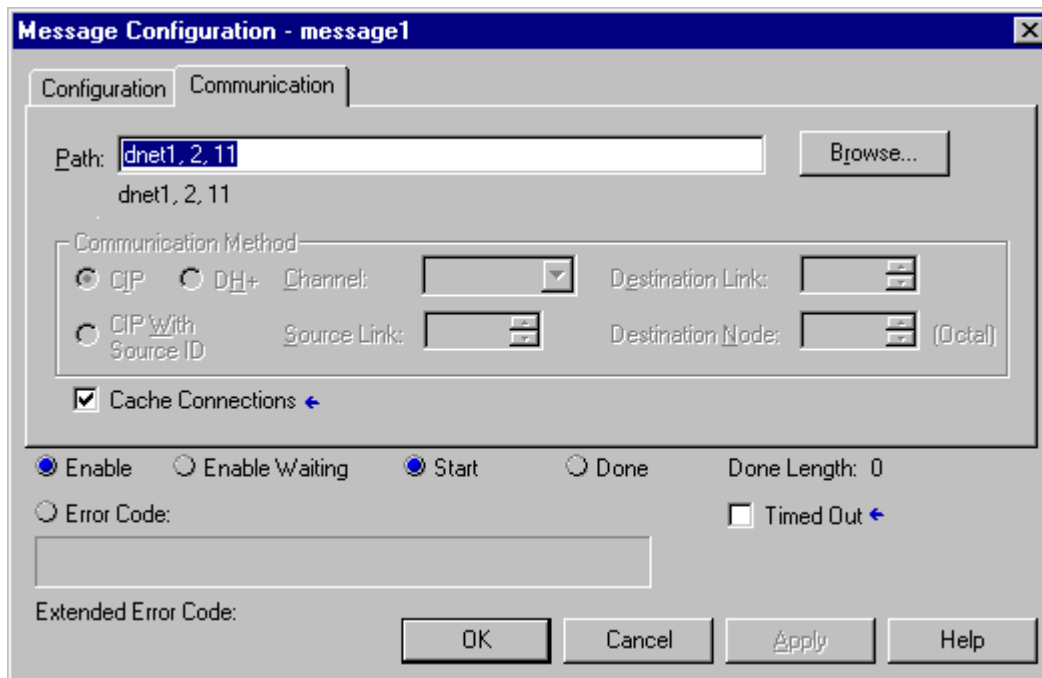
'Num. Of Elements' – bytes of additional service data to be sent or received in the message command.



'Destination' - tag for any additional service response data.

The values entered for the Service Code, Object Type, Object ID, and Object Attribute are dependent on the type of explicit message you would like to perform and are defined in the 1203-GU6 User Manual.

Figure 3 – Message Command Communication Configuration



Click on the 'Communication' tab to configure the communication path.

'Path' - 'name of DeviceNet scanner module', 'communication port on the front of the 1756-DNB module', 'node address of the GU6'.

' name of the DeviceNet scanner module' - can be found through the Browse button and is the same as the name of your 1756-DNB under the I/O configuration in RSLogix5000.

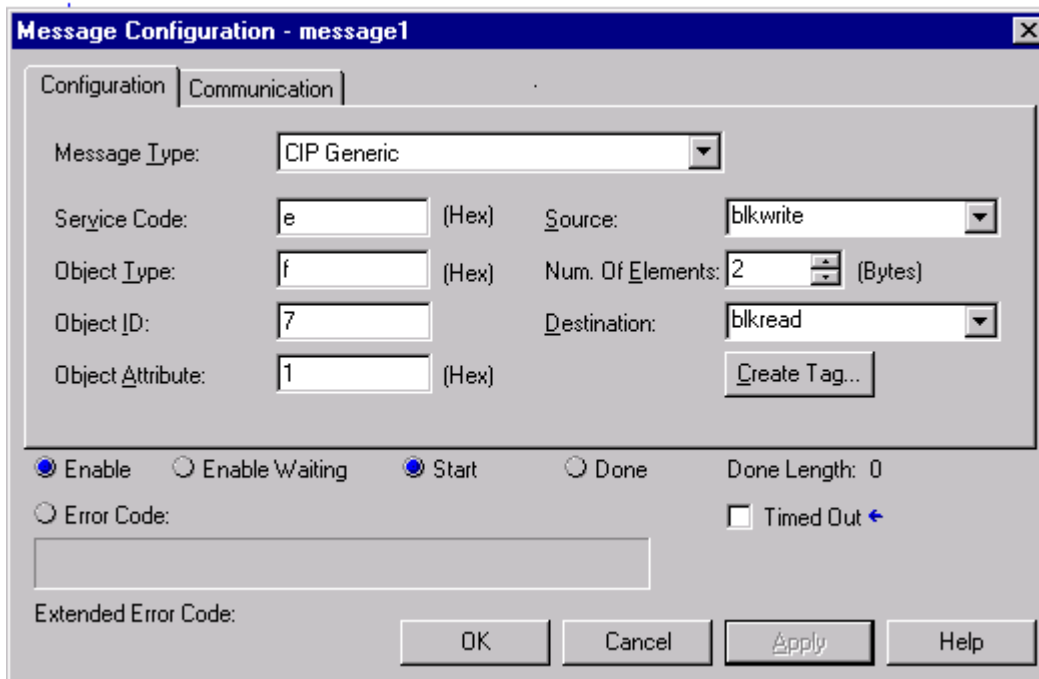
'communication port on the front of the 1756-DNB module' – always set to 2.

'node address of the GU6' - the DeviceNet node address of the GU6 module.



SINGLE PARAMETER VALUE READ EXAMPLE

Figure 4 – Single Parameter Value Read Message Command Configuration



For a single parameter read the 'Service Code' is E (get). The 'Object Type' is F (Parameter Object). The 'Object ID' is the number of the parameter we would like to read from the SCANport product. In this example we are reading parameter 7 [Accel Time 1] from a 1336 PLUS drive. The 'Object Attribute' is 1 (to read a Parameter Value).

The 'Communication' tab is setup the same as in figure 3 for this example. The GU6 with the 1336 PLUS drive is at node 11.

'blkwrite' is a tag created for an array of 16 integers. No additional source data will actually be sent in this example. The 'Num. of Elements' is 2 Bytes since 1 word will be read back from the SCANport device. The 'Destination' is for any additional service response data. 'blkread' is a tag created for an array of 16 integers. 'blkread[0]' will be read back as the value of parameter 7 as shown in figure 5.

Figure 5 – Single Parameter Value Read Tags

Tag Name	Value	Force Mask	Style	Type	Description
blkread	{...}	{...}	Decimal	INT[16]	
blkread[0]	400		Decimal	INT	
blkread[1]	0		Decimal	INT	
blkread[2]	0		Decimal	INT	



SCATTERED PARAMETER VALUE READ EXAMPLE

Figure 6 – Scattered Parameter Value Read Message Command Configuration

Message Configuration - message1

Configuration | Communication

Message Type: CIP Generic

Service Code: 32 (Hex) Source: blkwrite

Object Type: 93 (Hex) Num. Of Elements: 32 (Bytes)

Object ID: 0 Destination: blkread

Object Attribute: 0 (Hex) Create Tag...

Enable Enable Waiting Start Done Done Length: 0

Error Code: Timed Out

Extended Error Code:

OK Cancel Apply Help

The 'Service Code' for a scattered parameter read is 32. The 'Object Type' is 93. 'Object ID' and 'Object Attribute' are not used and are set to 0.

The 'Communication' tab will be set up the same as in Figure 3.

'blkwrite' is used to specify what parameters to read. Starting with 'blkwrite[0]', enter the parameter number to read in every other word of 'blkwrite'. 'blkread' reads back the parameter numbers, their values, and any error messages.



Figure 7 – Scattered Parameter Value Read Tags

Tag Name	Value	Force Mask	Style	Type	Description
- blkread	{...}	{...}	Decimal	INT[16]	
+ blkread[0]	1		Decimal	INT	
+ blkread[1]	0		Decimal	INT	
+ blkread[2]	7		Decimal	INT	
+ blkread[3]	100		Decimal	INT	
+ blkread[4]	8		Decimal	INT	
+ blkread[5]	100		Decimal	INT	
+ blkread[6]	17		Decimal	INT	
+ blkread[7]	600		Decimal	INT	
+ blkread[8]	-32768		Decimal	INT	
+ blkread[9]	4		Decimal	INT	
+ blkread[10]	-32768		Decimal	INT	
+ blkread[11]	4		Decimal	INT	
+ blkread[12]	-32768		Decimal	INT	
+ blkread[13]	4		Decimal	INT	
+ blkread[14]	-32768		Decimal	INT	
+ blkread[15]	4		Decimal	INT	
- blkwrite	{...}	{...}	Decimal	INT[16]	
+ blkwrite[0]	1		Decimal	INT	
+ blkwrite[1]	0		Decimal	INT	
+ blkwrite[2]	7		Decimal	INT	
+ blkwrite[3]	0		Decimal	INT	
+ blkwrite[4]	8		Decimal	INT	
+ blkwrite[5]	0		Decimal	INT	
+ blkwrite[6]	17		Decimal	INT	
+ blkwrite[7]	0		Decimal	INT	
+ blkwrite[8]	0		Decimal	INT	
+ blkwrite[9]	0		Decimal	INT	

Likely error responses are:

1	An attempt was made to write to a read only variable.
4	Parameter number is out of range (Instance not supported).
6	Data to be written was out of range.
7	Object State Conflict (e.g. an attempt to change a variable that is not changeable in the run state).



SCATTERED PARAMETER VALUE WRITE EXAMPLE

Using explicit messaging to make frequent changes to a parameter will eventually result in the failure of the SCANport device's EEPROM (if so equipped). If an application requires frequent changes of only a few parameters, the parameters should be written using the SCANport Datalink function (if available) since this does not cause EEPROM writes to occur.

Figure 8 – Scattered Parameter Value Write Message Command Configuration

The screenshot shows a 'Message Configuration' dialog box with two tabs: 'Configuration' and 'Communication'. The 'Communication' tab is selected. The 'Message Type' is set to 'CIP Generic'. The 'Service Code' is 34 (Hex), 'Object Type' is 93 (Hex), 'Object ID' is 0, and 'Object Attribute' is 0 (Hex). The 'Source' is 'blkwrite' and the 'Destination' is 'blkread'. The 'Num. Of Elements' is 32 (Bytes). There is a 'Create Tag...' button. Below the fields are radio buttons for 'Enable', 'Enable Waiting', 'Start', and 'Done', with 'Start' selected. There is also a 'Timed Out' checkbox. At the bottom are buttons for 'OK', 'Cancel', 'Apply', and 'Help'.

For the scattered parameter write the 'Service Code' is 34. The 'Object Type' is 93. 'Object ID' and 'Object Attribute' are not used and are set to 0.

'blkwrite' contains the parameters that you wish to write and their values. 'blkread' contains the parameter number you wrote and any error messages. The 'Communication' tab will be setup the same as in Figure 3.



Figure 9 – Scattered Parameter Value Write Tags

Scope: josh(controller) Shgw: Show All Sort: Tag Name							
Tag Name	Value	Force Mask	Style	Type	Description		
- blkread	{...}	{...}	Decimal	INT[16]			
+ blkread[0]	16		Decimal	INT			
+ blkread[1]	0		Decimal	INT			
+ blkread[2]	19		Decimal	INT			
+ blkread[3]	0		Decimal	INT			
+ blkread[4]	7		Decimal	INT			
+ blkread[5]	0		Decimal	INT			
+ blkread[6]	8		Decimal	INT			
+ blkread[7]	0		Decimal	INT			
+ blkread[8]	-32768		Decimal	INT			
+ blkread[9]	4		Decimal	INT			
+ blkread[10]	-32768		Decimal	INT			
+ blkread[11]	4		Decimal	INT			
+ blkread[12]	-32768		Decimal	INT			
+ blkread[13]	4		Decimal	INT			
+ blkread[14]	-32768		Decimal	INT			
+ blkread[15]	4		Decimal	INT			
- blkwrite	{...}	{...}	Decimal	INT[16]			
+ blkwrite[0]	16		Decimal	INT			
+ blkwrite[1]	100		Decimal	INT			
+ blkwrite[2]	19		Decimal	INT			
+ blkwrite[3]	900		Decimal	INT			
+ blkwrite[4]	7		Decimal	INT			
+ blkwrite[5]	200		Decimal	INT			
+ blkwrite[6]	8		Decimal	INT			
+ blkwrite[7]	50		Decimal	INT			
+ blkwrite[8]	0		Decimal	INT			
+ blkwrite[9]	0		Decimal	INT			