

1203-GK5 & 1336-GM5 WITH 1747-SDN OR 1771-SDN EXPLICIT MESSAGING FOR FAULTS AND WARNINGS

APPLICATION NOTE

OCTOBER 16, 1997

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.

WHAT THIS NOTE CONTAINS

This document contains information and ladder program examples showing how to perform explicit messaging to obtain fault and warning information from SCANport products via DeviceNet. Examples are shown for both 1771-SDN and 1747-SDN scanners.

INTENDED AUDIENCE

This application note should be used by personnel familiar with the hardware components and programming procedures necessary to operate DeviceNet and SCANport devices. It is also assumed that the user has some familiarity with the equipment used and with 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.

TERMS AND DEFINITIONS

BTR — Block Transfer Read

BTW — Block Transfer Write

Datalink — a pointer used by some SCANport products to allow parameters to be mapped to adapter I/O.

EEPROM — the memory that stores drive parameters when power is not applied.

Explicit Messaging — a DeviceNet messaging standard.

Fault Queue — a portion of memory inside a SCANport product that stores information about faults.

Warning Queue — a portion of memory inside a SCANport product that stores information about warnings.

Fault Queue Entry — a record of a particular fault occurrence stored within the fault queue.

Warning Queue Entry — a record of a particular warning occurrence stored within the warning queue.

Fault Queue Trip Index — the number identifying the fault queue entry containing information about a fault that caused the drive to trip.

Fault Command — a method of instructing the drive to perform a function on the fault queue.

Warning Command — a method of instructing the drive to perform a function on the warning queue.

DESCRIPTION

For clarity, only one DeviceNet explicit message transaction is active at any time in these examples.

The first example uses a PLC-5, a 1771-SDN DeviceNet scanner and a 1203-GK5 to read a fault queue entry in a SCANport product. The program will execute a BTW and then execute multiple BTR's until the DeviceNet message completes.

The second example uses an SLC-5/03, a 1747-SDN DeviceNet scanner and a 1203-GK5 to read a fault queue entry in a SCANport product. The program will write a message into the scanner and then read the response from the scanner when the DeviceNet message completes. The program then deletes the completed message from the scanner queue.

Complex applications may require additional logic to poll the DeviceNet scanner to check for message completion and may have multiple active transactions.

APPLICATION CONSIDERATIONS

These ladder programs were written to be simple and clear examples of DeviceNet messaging. They contain no error handling. Consult the SLC-500, PLC-5, 1771-SDN, 1747-SDN and 1203-GK5 manuals for more information.

Explicit messages will complete faster if the scanner is first placed in the idle mode. This may be worthwhile if the application requires reading or writing a large number of parameters in the SCANport product (e.g., configuring a drive system for a different product).

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

SCANport FAULT OBJECT

The SCANport Fault Object is a DeviceNet Vendor Specific Object with a class code of 151 (97h).

Service	Class	Instance	Attribute	Data	Description
10h	97h	0	0	d	Write Fault Command 1=Clear Faults 2=Clear Fault Queue 3=Reset Product ²
0Eh	97h	0	1		Read Number of Fault Queue Entries
0Eh	97h	0	2		Read Fault Queue Trip Index
0Eh	97h	i ¹	0		Read Fault Queue Entry Full/All Info
0Eh	97h	i ¹	128 ²		Read Fault Code and Time Stamp ²
0Eh	97h	i ¹	129 ²		Read Fault Text String Only ²

¹ i = Index number of the Fault Queue Entry.

² Optional Value – not all SCANport products support this function.

³ Optional Attribute – not all SCANport products support this function.

SCANport WARNING OBJECT

The SCANport Warning Object is a DeviceNet Vendor Specific Object with a class code of 152 (98h). Support for this object is optional – not all SCANport products will support these functions.

10h	98h	0	0	d	Write Warning Command	1=Clear Warnings 2=Clear Warning Queue 3=Reset Product ²
0Eh	98h	0	1		Read Number of Warning Queue Entries	
0Eh	98h	i ¹	0		Read Warning Queue Entry Full/All Info	
0Eh	98h	i ¹	128 ²		Read Warning Code and Time Stamp ²	
0Eh	98h	i ¹	129 ²		Read Warning Text String Only ²	

- ¹ i = Index number of the Warning Queue Entry.
- ² Optional Value – not all SCANport products support this function.
- ³ Optional Attribute – not all SCANport products support this function.

1771-SDN — HOW TO FORMAT THE EXPLICIT MESSAGE TRANSACTION BLOCK

Ten 32-word transaction blocks within the scanner module are reserved for Explicit Message Program Control. The transaction blocks accommodate both the download of Explicit Message Requests and the upload of Explicit Message Responses.

The scanner module can accommodate one request or response for each transaction block and can transfer two blocks for each upload and download. You must format each transaction block as shown:

		Format of 64-word Block Transfer Write for Explicit Message Request		Format of 64-word Block Transfer Read for Explicit Message Response		
		15	0	15	0	
Transaction #1 Header (3 words)		TXID	COMMAND	TXID	STATUS	word 0
		PORT	SIZE	PORT	SIZE	
		SERVICE	MAC ID	SERVICE	MAC ID	
		CLASS		SERVICE RESPONSE DATA		
		INSTANCE		"		
		ATTRIBUTE		"		
		SERVICE DATA		"		
		"		"		word 31
		"		"		word 32
Transaction #2 Header (3 words)		TXID	COMMAND	TXID	STATUS	
		PORT	SIZE	PORT	SIZE	
		SERVICE	MAC ID	SERVICE	MAC ID	
		CLASS		SERVICE RESPONSE DATA		
		INSTANCE		"		
		ATTRIBUTE		"		
		SERVICE DATA		"		
		"		"		word 63
		"		"		

Allen-Bradley

Transaction Blocks are divided into two parts:

- **transaction header** — contains information that identifies the transaction to the scanner and processor
- **transaction body** — in a request, this contains the DeviceNet Class, Instance, Attribute and Service Data portion of the transaction. In a response, this contains only the response message.

Each of the data attributes in the transaction header are one byte in length:

- **COMMAND** — for each download, a command code instructs the scanner how to administer the request:

Command Code	Description
0	Ignore transaction block (block empty)
1	Execute this transaction block
2	Get status of transaction TXID
3	Reset all client/server transactions
4-255	Reserved

- **STATUS** — for each upload, the status code provides the processor with status on the device and its response:

Status Code	Description
0	Ignore transaction block (block empty)
1	Transaction completed successfully
2	Transaction in progress (not ready)
3	Error — slave not in scan list
4	Error — slave off-line
5	Error — DeviceNet port disabled or off-line
6	Error — transaction TXID unknown
7	Unused
8	Error — Invalid command code
9	Error — Scanner out of buffers
10	Error — Other client/server transaction in progress
11	Error — could not connect to slave device
12	Error — response data too large for block
13	Error — invalid port
14	Error — invalid size specified
15	Error — connection busy
16-255	Reserved

Allen-Bradley

- **TXID** Transaction ID — when you create and download a request to the scanner, the processor's ladder logic program assigns a TXID to the transaction. This is a one-byte integer in word 31 the range of 1 to 255. The scanner uses this value to track the transaction to completion, and returns the value with the response that matches the request downloaded by the processor. The ladder logic program monitors rollover and usage of TXID values.
- **SIZE** The size of the transaction body in bytes. The transaction body can be up to 29 words (58 bytes) in length. If the size exceeds 29 words, an error code will be returned.
- **PORT** The DeviceNet port where the transaction is routed. The port can be zero (Channel A) or one (Channel B).
- **MAC ID** The DeviceNet network address of the slave device where the transaction is sent. This value can range from 0 to 63. The port and MAC ID attributes coupled together identify the target slave device. The slave device must be listed in the scanner module's scan list and be on-line for the Explicit Message transaction to be completed.
- **SERVICE** The service attribute contains the service request and response codes that match the corresponding request for the TXID.

1771-SDN — HOW THE PROCESSOR AND SCANNER MODULE MANAGE MESSAGES

Block transfer operations between the processor and the scanner always originate in the processor. The scanner module can only wait for the processor to download a transaction block to the module or request an upload of a transaction block from the module.

Once an Explicit Message Request transaction block is downloaded to the scanner module, a ladder logic program in the processor polls the scanner module for the transaction block containing the Explicit Message Response for that request. This is done by the processor with a Block Transfer Read on the scanner module. Depending on the network load, the scanner could take a few seconds to complete the request. When a response is loaded, bit 15 of the module status register is set to 1. The program may have to poll the scanner module a number of times before the scanner returns a Response Transaction Block.

1771-SDN — INTERLOCKED BTW/BTR PARAMETER READ AND WRITE EXAMPLE

Figure 14 shows one method of interlocking the block transfer write and read functions so that a message to the 1203-GK5 is automatically monitored for a response. This program uses the same data table values as shown in Figures 2 and 3.

When the value in N21:70 matches the value in N21:0 the message has successfully completed. At that time, the data table contains the response message and another message can be sent to the 1203-GK5. If the value of the low byte of N21:70 is anything other than 0, 1 or 2 an error has occurred and other error handling should be performed.

```

I:001  B3
--] [--[ONS]-----+BTW-----+
    00    0          |BLOCK TRANSFER WRITE  +-(EN)-
                    |Rack                    00|
                    |Group                   0+-(DN)|
                    |Module                   1|
                    |Control block          BT20:0+-(ER)|
                    |Data file              N21:0|
                    |Length                 64|
                    |Continuous             N|
                    +-----+
                    +MOV-----+
                    |MOVE                    ++
                    |Source                   0|
                    |Destination N21:70
                    |                       257|
                    +-----+

BT20:0 +CMP-----+ BT20:1          +BTR-----+
--] [---|COMPARE      +---]/[-----+BLOCK TRANSFER READ  +-(EN)-
    DN  |Expression    |      EN          |Rack                    00|
        |N21:70 <> N21:0|      |          |Group                   0+-(DN)|
        +-----+          |          |Module                   1|
                    |Control block          BT20:1+-(ER)|
                    |Data file              N21:70|
                    |Length                 64|
                    |Continuous             N|
                    +-----+

```

1771-SDN — Interlocked BTW/BTR Example

1771-SDN DATA FILES

Offset	0	1	2	3	4	5	6	7	8	9
N21:0	101	6	1001	97	0	1	0	0	0	0
N21:10	0	0	0	0	0	0	0	0	0	0
N21:20	0	0	0	0	0	0	0	0	0	0
N21:30	0	0	0	0	0	0	0	0	0	0
N21:40	0	0	0	0	0	0	0	0	0	0
N21:50	0	0	0	0	0	0	0	0	0	0
N21:60	0	0	0	0	0	0	0	0	0	0
N21:70	101	0	9001	0	0	0	0	0	0	0
N21:80	0	0	0	0	0	0	0	0	0	0
N21:90	0	0	0	0	0	0	0	0	0	0
N21:100	0	0	0	0	0	0	0	0	0	0
N21:110	0	0	0	0	0	0	0	0	0	0
N21:120	0	0	0	0	0	0	0	0	0	0
N21:130	0	0	0	0						

1771-SDN — Write Fault Command = 1 (Clear Faults)

Allen-Bradley

Offset	0	1	2	3	4	5	6	7	8	9
N21:0	101	6	1001	97	0	2	0	0	0	0
N21:10	0	0	0	0	0	0	0	0	0	0
N21:20	0	0	0	0	0	0	0	0	0	0
N21:30	0	0	0	0	0	0	0	0	0	0
N21:40	0	0	0	0	0	0	0	0	0	0
N21:50	0	0	0	0	0	0	0	0	0	0
N21:60	0	0	0	0	0	0	0	0	0	0
N21:70	101	0	9001	0	0	0	0	0	0	0
N21:80	0	0	0	0	0	0	0	0	0	0
N21:90	0	0	0	0	0	0	0	0	0	0
N21:100	0	0	0	0	0	0	0	0	0	0
N21:110	0	0	0	0	0	0	0	0	0	0
N21:120	0	0	0	0	0	0	0	0	0	0
N21:130	0	0	0	0						

1771-SDN — Write Fault Command = 2 (Clear Fault Queue)

Offset	0	1	2	3	4	5	6	7	8	9
N21:0	101	6	E01	97	0	1	0	0	0	0
N21:10	0	0	0	0	0	0	0	0	0	0
N21:20	0	0	0	0	0	0	0	0	0	0
N21:30	0	0	0	0	0	0	0	0	0	0
N21:40	0	0	0	0	0	0	0	0	0	0
N21:50	0	0	0	0	0	0	0	0	0	0
N21:60	0	0	0	0	0	0	0	0	0	0
N21:70	101	6	8E01	4	0	0	0	0	0	0
N21:80	0	0	0	0	0	0	0	0	0	0
N21:90	0	0	0	0	0	0	0	0	0	0
N21:100	0	0	0	0	0	0	0	0	0	0
N21:110	0	0	0	0	0	0	0	0	0	0
N21:120	0	0	0	0	0	0	0	0	0	0
N21:130	0	0	0	0						

1771-SDN — Read Number of Fault Queue Entries = 4

Offset	0	1	2	3	4	5	6	7	8	9
N21:0	101	6	E01	97	0	2	0	0	0	0
N21:10	0	0	0	0	0	0	0	0	0	0
N21:20	0	0	0	0	0	0	0	0	0	0
N21:30	0	0	0	0	0	0	0	0	0	0
N21:40	0	0	0	0	0	0	0	0	0	0
N21:50	0	0	0	0	0	0	0	0	0	0
N21:60	0	0	0	0	0	0	0	0	0	0
N21:70	101	6	8E01	1	0	0	0	0	0	0
N21:80	0	0	0	0	0	0	0	0	0	0
N21:90	0	0	0	0	0	0	0	0	0	0
N21:100	0	0	0	0	0	0	0	0	0	0
N21:110	0	0	0	0	0	0	0	0	0	0
N21:120	0	0	0	0	0	0	0	0	0	0
N21:130	0	0	0	0						

1771-SDN — Read Trip Fault Queue Entry Index = 1

Allen-Bradley

Offset	0	1	2	3	4	5	6	7	8	9
N21:0	101	6	E01	97	1	0	0	0	0	0
N21:10	0	0	0	0	0	0	0	0	0	0
N21:20	0	0	0	0	0	0	0	0	0	0
N21:30	0	0	0	0	0	0	0	0	0	0
N21:40	0	0	0	0	0	0	0	0	0	0
N21:50	0	0	0	0	0	0	0	0	0	0
N21:60	0	0	0	0	0	0	0	0	0	0
N21:70	101	1E	8E01	6553	6972	6C61	4620	7561	746C	2020
N21:80	2020	0	0	0	0	0	0	0	0	0
N21:90	0	0	0	0	0	0	0	0	0	0
N21:100	0	0	0	0	0	0	0	0	0	0
N21:110	0	0	0	0	0	0	0	0	0	0
N21:120	0	0	0	0	0	0	0	0	0	0
N21:130	0	0	0	0						

1771-SDN — Read Fault Queue Entry 1 = "Serial Fault "

Offset	0	1	2	3	4	5	6	7	8	9
N21:0	\01\01	\00\06	\0E\01	\00\97	\00\01	\00\00	\00\00	\00\00	\00\00	\00\00
N21:10	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N21:20	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N21:30	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N21:40	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N21:50	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N21:60	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N21:70	\01\01	\00\1E	\8E\01	es	ir	la	F	ua	tl	
N21:80		\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N21:90	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N21:100	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N21:110	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N21:120	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N21:130	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00

1771-SDN — Read Fault Queue Entry 1 (in ASCII) = "Serial Fault "

1747-SDN — HOW TO FORMAT THE EXPLICIT MESSAGE TRANSACTION BLOCK

Ten 32-word transaction blocks within the scanner module are reserved for Explicit Message Program Control. The transaction blocks accommodate both the download of Explicit Message Requests and the upload of Explicit Message Responses.

The scanner module can accommodate one request or response for each transaction block and can transfer two blocks for each upload and download. You must format each transaction block as shown:

		Format of 31-word M0-file Write for Explicit Message Request		Format of 31-word M1-file Read for Explicit Message Response		
		15	0	15	0	
Transaction #1		TXID	COMMAND	TXID	STATUS	word 0
Header		PORT	SIZE	PORT	SIZE	
(3 words)		SERVICE	MAC ID	SERVICE	MAC ID	
		CLASS		SERVICE RESPONSE DATA		
Transaction		INSTANCE		"		
Data		ATTRIBUTE		"		
(up to 29)		SERVICE DATA		"		
(words)		"		"		word 31

Transaction Blocks are divided into two parts:

- **transaction header** — contains information that identifies the transaction to the scanner and processor
- **transaction body** — in a request, this contains the DeviceNet Class, Instance, Attribute and Service Data portion of the transaction. In a response, this contains only the response message.

Each of the data attributes in the transaction header are one byte in length:

- **COMMAND** — for each download, a command code instructs the scanner how to administer the request:

Command Code	Description
0	Ignore transaction block (block empty)
1	Execute this transaction block
2	Get status of transaction TXID
3	Reset all client/server transactions
4	Delete this transaction block
5-255	Reserved

Allen-Bradley

- **STATUS** — for each upload, the status code provides the processor with status on the device and its response:

Status Code	Description
0	Ignore transaction block (block empty)
1	Transaction completed successfully
2	Transaction in progress (not ready)
3	Error — slave not in scan list
4	Error — slave off-line
5	Error — DeviceNet port disabled or off-line
6	Error — transaction TXID unknown
7	Unused
8	Error — Invalid command code
9	Error — Scanner out of buffers
10	Error — Other client/server transaction in progress
11	Error — could not connect to slave device
12	Error — response data too large for block
13	Error — invalid port
14	Error — invalid size specified
15	Error — connection busy
16-255	Reserved

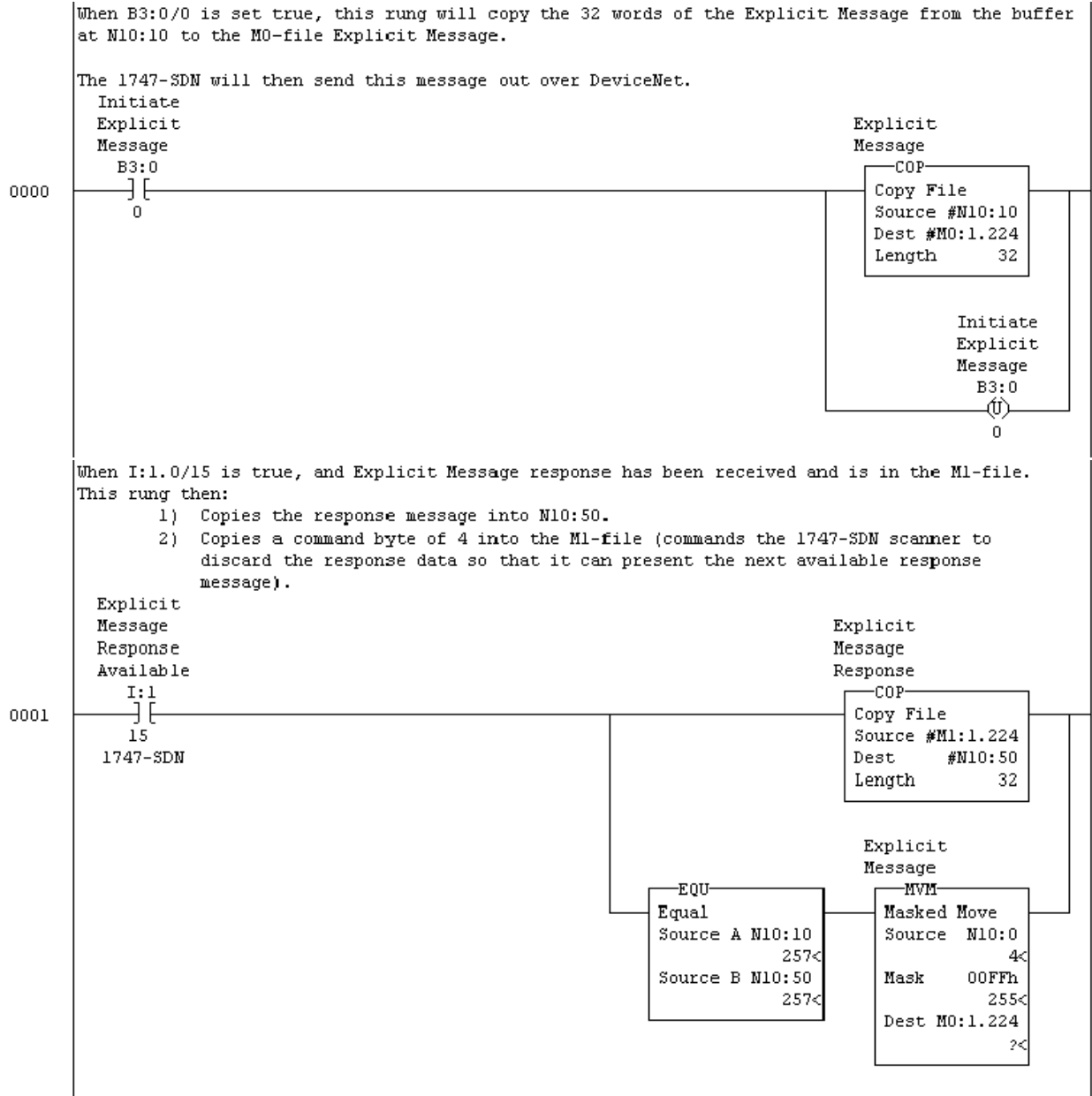
- **TXID** Transaction ID — when you create and download a request to the scanner, the processor's ladder logic program assigns a TXID to the transaction. This is a one-byte integer in word 31 the range of 1 to 255. The scanner uses this value to track the transaction to completion, and returns the value with the response that matches the request downloaded by the processor. The ladder logic program monitors rollover and usage of TXID values.
- **SIZE** The size of the transaction body in bytes. The transaction body can be up to 29 words (58 bytes) in length. If the size exceeds 29 words, an error code will be returned.
- **PORT** The DeviceNet port where the transaction is routed. The port can be zero (Channel A) or one (Channel B).
- **MAC ID** The DeviceNet network address of the slave device where the transaction is sent. This value can range from 0 to 63. The port and MAC ID attributes coupled together identify the target slave device. The slave device must be listed in the scanner module's scan list and be on-line for the Explicit Message transaction to be completed.
- **SERVICE** The service attribute contains the service request and response codes that match the corresponding request for the TXID.

1747-SDN — HOW THE PROCESSOR AND SCANNER MODULE MANAGE MESSAGES

The SLC copies an Explicit Message into the scanner's M0-file. When the copy is completed the scanner moves the message into a queue for processing. Up to 10 Explicit Messages can be in this queue.

When the scanner receives a response message it is placed into a queue. The first response in the queue is available from the M1-file. When the message delete command is copied into the scanner the message cycle is complete and the next available response will appear in the M1-file.

EXPLICIT MESSAGING EXAMPLE LADDER PROGRAM



1747-SDN DATA FILES

Offset	0	1	2	3	4	5	6	7	8	9
N10:0	4	0	0	0	0	0	0	0	0	0
N10:10	101	6	1001	97	0	1	0	0	0	0
N10:20	0	0	0	0	0	0	0	0	0	0
N10:30	0	0	0	0	0	0	0	0	0	0
N10:40	0	0	0	0	0	0	0	0	0	0
N10:50	101	0	9001	0	0	0	0	0	0	0
N10:60	0	0	0	0	0	0	0	0	0	0
N10:70	0	0	0	0	0	0	0	0	0	0
N10:80	0	0								

1747-SDN — Write Fault Command = 1 (Clear Faults)

Offset	0	1	2	3	4	5	6	7	8	9
N10:0	4	0	0	0	0	0	0	0	0	0
N10:10	101	6	1001	97	0	2	0	0	0	0
N10:20	0	0	0	0	0	0	0	0	0	0
N10:30	0	0	0	0	0	0	0	0	0	0
N10:40	0	0	0	0	0	0	0	0	0	0
N10:50	101	0	9001	0	0	0	0	0	0	0
N10:60	0	0	0	0	0	0	0	0	0	0
N10:70	0	0	0	0	0	0	0	0	0	0
N10:80	0	0								

1747-SDN — Write Fault Command = 2 (Clear Fault Queue)

Offset	0	1	2	3	4	5	6	7	8	9
N10:0	4	0	0	0	0	0	0	0	0	0
N10:10	101	6	E01	97	0	1	0	0	0	0
N10:20	0	0	0	0	0	0	0	0	0	0
N10:30	0	0	0	0	0	0	0	0	0	0
N10:40	0	0	0	0	0	0	0	0	0	0
N10:50	101	6	8E01	4	0	0	0	0	0	0
N10:60	0	0	0	0	0	0	0	0	0	0
N10:70	0	0	0	0	0	0	0	0	0	0
N10:80	0	0								

1747-SDN — Read Number of Fault Queue Entries = 4

Offset	0	1	2	3	4	5	6	7	8	9
N10:0	4	0	0	0	0	0	0	0	0	0
N10:10	101	6	E01	97	0	2	0	0	0	0
N10:20	0	0	0	0	0	0	0	0	0	0
N10:30	0	0	0	0	0	0	0	0	0	0
N10:40	0	0	0	0	0	0	0	0	0	0
N10:50	101	6	8E01	1	0	0	0	0	0	0
N10:60	0	0	0	0	0	0	0	0	0	0
N10:70	0	0	0	0	0	0	0	0	0	0
N10:80	0	0								

1747-SDN — Read Trip Fault Queue Entry Index = 1

Allen-Bradley

Offset	0	1	2	3	4	5	6	7	8	9
N10:0	4	0	0	0	0	0	0	0	0	0
N10:10	101	6	E01	97	1	0	0	0	0	0
N10:20	0	0	0	0	0	0	0	0	0	0
N10:30	0	0	0	0	0	0	0	0	0	0
N10:40	0	0	0	0	0	0	0	0	0	0
N10:50	101	1E	8E01	6553	6972	6C61	4620	7561	746C	2020
N10:60	2020	0	0	0	0	0	0	0	0	0
N10:70	0	0	0	0	0	0	0	0	0	0
N10:80	0	0								

1747-SDN — Read Fault Queue Entry 1 = "Serial Fault "

Offset	0	1	2	3	4	5	6	7	8	9
N10:0	\00\04	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N10:10	\01\01	\00\06	\0E\01	\00\97	\00\01	\00\00	\00\00	\00\00	\00\00	\00\00
N10:20	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N10:30	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N10:40	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N10:50	\01\01	\00\1E	\8E\01	eS	ir	la	F	ua	t1	
N10:60		\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N10:70	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00
N10:80	\00\00	\00\00								

1747-SDN — Read Fault Queue Entry 1 (in ASCII) = "Serial Fault "