Date: June 14, 2004

To: T10 Committee (SCSI)

From: George Penokie (IBM/Tivoli)

Subject: SPC-3: Standard Data Collection Trigger

1 Overview

Debugging a problem in a SCSI environment can be difficult when one part of the system (e.g., application client) detects an error when the failure is caused by a different part of the system (e.g., logical unit).

Most applications and logical units have some of error logging, but correlating the time of an application client event with the time of a logical unit log can be difficult.

Correlating the location of data in a log can be difficult. For example, the application client may complain about "scsi disk 0" while the logical unit only knows a about it's LUN.

In logical units recoverable errors typically generate minimal log information, while catastrophic errors often generate detailed engineering data. A problem that may be viewed as recoverable at the logical unit may cause a catastrophic failure at the application client. Also, there is no general way to automatically trigger collection of more detailed log data in a logical unit.

This proposal defines a way to pass error information to a logical unit. The information would not be used for any error recovery procedures, but would be used in to enhance debug. For example, the logical unit would add the information received from the application client to its existing error log. The error log could then be used to correlate an application client error with any errors internal to the logical unit. This doesn't replace the vendor specific methods for collecting and analyzing engineering data, but provides a vendor independent way of correlating error logs and provides a trigger mechanism for collecting additional log data.

2 Solution

2.1 WRITE BUFFER command

2.1.1 WRITE BUFFER command introduction

The WRITE BUFFER command (see table 1) is used in conjunction with the READ BUFFER command as a diagnostic function for testing logical unit memory in the target SCSI device and the integrity of the service delivery subsystem. Additional modes are provided for:

a) downloading microcode:
b) for downloading and saving microcode: and

c) Downloading application client logs.

Table 1 — WRITE BUFFER command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (3Bh)							
1	Reserved			MODE				
2	BUFFER ID							
3	(MSB)							
4				BUFFER OFFSET				
5								(LSB)
6	(MSB)							
7				PARAMETER LIST LENGTH				
8								(LSB)
9	CONTROL							

This command shall not alter any medium of the logical unit when the data mode or the combined header and data mode is specified.

The function of this command and the meaning of fields within the CDB depend on the contents of the MODE field. The MODE field is defined in table 77.

Table 2 — WRITE BUFFER MODE field

MODE	Description	Implementation requirements
00h	Write combined header and data	Optional
01h	Vendor specific	Vendor specific
02h	Write data	Optional
04h	Download microcode	Optional
05h	Download microcode and save	Optional
06h	Download microcode with offsets	Optional
07h	Download microcode with offsets and save	Optional
0Ah	Echo buffer	Optional
1Ah	Enable expander communications protocol and Echo buffer	Optional
1Bh	Disable expander communications protocol	Optional
1Ch	Download application log	Optional
03h	Reserved	
08h - 09h	Reserved	
0Bh - 19h	Reserved	
1Dh - 1Fh	Reserved	

NOTES
1 Modes 00h and0 001h are not recommended.
2 When downloading microcode with buffer offsets, the WRITE BUFFER command mode should be 06h or 07h.

2.1.2 Combined header and data mode (00h)

....

2.1.3 Vendor specific mode (01h)

....

2.1.4 Data mode (02h)

....

2.1.5 Download microcode mode (04h)

....

2.1.6 Download microcode and save mode (05h)

....

2.1.7 Download microcode with offsets (06h)

....

2.1.8 Download microcode with offsets and save mode (07h)

....

2.1.9 Write data to echo buffer (0Ah)

....

2.1.10 Enable expander communications protocol and Echo buffer (1Ah)

....

2.1.11 Disable expander communications protocol (1Bh)

....

2.1.12 Download application log (1Ch) (new section)

In this mode the device server transfers data from the application client and stores it in a log. The format of the log data is as specified in table 3. The BUFFER ID and BUFFER OFFSET fields are ignored in this mode.

Upon successful completion of a WRITE BUFFER command the data shall be preserved in a log.

The PARAMETER LIST LENGTH field specifies the maximum number of bytes that shall be transferred from the Data-Out Buffer to be stored in the log. If the PARAMETER LIST LENGTH field specifies a transfer in excess of the

logs capacity, the device server shall return CHECK CONDITION status and shall set the sense key to ILLEGAL REQUEST with an additional sense code of INVALID FIELD IN CDB.

Table 3 — Log data format

Bit Byte	7	6	5	4	3	2	1	0
0	Reserved							
1	Reserved							
2	Reserved							
3	Reserved							
4	(MSB)							
7				ERROR TYPE				(LSB)
8								
15				TIME STAMP				
16	Reserved				CODE SET			
17	ERROR LOCATION FORMAT							
18	Reserved							
19	Reserved							
20	(MSB)							
21				ERROR LOCATION LENGTH (24-m)				(LSB)
22	(MSB)							
23				VENDOR SPECIFIC LENGTH (n-m)				(LSB)
24	(MSB)							
m				ERROR LOCATION				(LSB)
m+1								
n				VENDOR SPECIFIC				

The ERROR TYPE field (see table 4) specifies the error detected by the application client.

Table 4 — ERROR TYPE field values

Value	Description
0000h	No error specified by the application client
0001h	Any unknown error was detected by the application client
0002h	The application client detected corrupted data
0003h	The application client detected a permanent error
0004h	The application client detected an I_T nexus failure
0005h - 00FFh	Reserved
0100h - FFFFh	Vendor specific

The TIME STAMP field is represented as a 64-bit unsigned fixed-point number in big-endian format. The time stamp is divided into two 32 bit fields (see table 5). They are the INTEGER field and the FRACTIONAL field.

Table 5 — TIME STAMP field format

Bit Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				INTEGER				(LSB)
4	(MSB)							
7				FRACTIONAL				(LSB)

The INTEGER field specifies the number of seconds relative to the zero hour on January 1, 1900 UTC. The maximum number of seconds represented is 4 294 967 296 (i.e., the year 2036).

The FRACTIONAL field specifies a fraction of a second (i.e. at value of 0001h is equal to 233 picoseconds). When the reference clock is not capable of such resolution, the non-significant low order bits may be padded with zeros.

The CODE SET field specifies the code set used for the IDENTIFIER field, as described in table 6. This field is intended to be an aid to software that displays the IDENTIFIER field.

Table 6 — Code set

Value	Description
0h	Reserved
1h	The IDENTIFIER field shall contain binary values.
2h	The IDENTIFIER field shall contain ASCII graphic codes (i.e., code values 20h through 7Eh)
3h	The IDENTIFIER field shall contain ISO/IEC 10646-1 (UTF-8) codes
4h - Fh	Reserved

The ERROR LOCATION FORMAT field specifies the format (see table 7) of the ERROR LOCATION field.

Table 7 — ERROR LOCATION FORMAT field values

Value	Description
00h	No error specified by the application client
01h	The ERROR LOCATION field specifies the LBA associated with the error information contained within the application log data.
02h - 7Fh	Reserved
80h - FFh	Vendor specific

The ERROR LOCATION LENGTH field specifies the length of the ERROR LOCATION field. The ERROR LOCATION LENGTH field value shall be a multiple of four. An error location length value of zero indicates there is no error location information.

The VENDOR SPECIFIC LENGTH field specifies the length of the VENDOR SPECIFIC field. The VENDOR SPECIFIC LENGTH field value shall be a multiple of four. A vendor specific length value of zero indicates there is no vendor specific information.

The ERROR LOCATION field specifies the location at which the application client detected the error.

The VENDOR SPECIFIC field provides vendor specific information on the error.