To: T10 Technical Committee
From: Rob Elliott, Compaq Computer Corporation (Robert.Elliott@compaq.com)
Date: 1 November 2000
Subject: Bidirectional data transfers in SPI-4

T10/00-309 by George Penokie proposes SAM modifications to allow protocols and commands to support bidirectional data transfers – commands which transfer both read and write data. This proposal modifies SPI-4 to support such transfers.

History

Revision 0, 11 August 2000: first revision

Revision 1, 23 August 2000:
Added type codes in SPI L_Q indicating direction (section 14.3.2). Changed “primary/secondary” terminology to “<nothing>/bidirectional read.”

Revision 2, 28 September 2000:
Changed MODIFY BIDIRECTIONAL READ DATA POINTER containing one pointer to MODIFY BIDIRECTIONAL POINTERS containing both pointers. This complicates error handling a bit – if one of the pointer moves is out of range, both are rejected.

The September Parallel SCSI WG recommended only supporting bidirectional commands in packetized mode, but I have chosen not to do that. The device would have to reject the commands based on negotiated mode, possibly confusing software. The INQUIRY CmdDT commands supported query cannot distinguish between negotiated settings, either.

Changed the Bidirectional Read and Bidirectional Read Stream type codes into a separate DIRECTION field in the L_Q that is only valid during Read and Read Stream type codes. This introduces an error case that does not exist if new type codes are used - the field could be incorrectly set in a command or status IU.

Added separate Data-In/Out buffer size to object call to match SAM-2 revision 14.

Revision 3, 1 November 2000
Reflecting a comment from Bill Galloway (Brea Technology) after the Parallel SCSI WG recommended revision 2, added text indicating that the target must disconnect before changing direction in non-IU mode. The text is in the SPI pointers section. “If information units are disabled, the target shall do a physical disconnect and physical reconnect to change the data transfer direction.”

Overview

The required changes mainly deal with the task data pointers. The initiator and target must maintain separate data pointers for the read and write streams. The MODIFY DATA POINTER message needs to be split into two, as any pointer arithmetic will have only one direction in mind. There are no reserved bits in the message to specify which pointer to use, so a new message was created. The SAVE DATA POINTER message can remain as is.

The SPI L_Q needs to indicate direction so the initiator can assume the correct state (pick the correct data-in or data-out pointer and start the appropriate DMA engine) before data starts flowing.

Changes are presented in order of SPI-4. The key changes are in sections 14.3.2, 15 and 16.3.8.

Global change
Change “SAVE DATA POINTER” to SAVE DATA POINTERS”.

14.1 SPI information unit overview
When a data transfer agreement is in effect that enables information unit transfers there is no option equivalent to the "physical disconnect without sending a SAVE DATA POINTERS message". The initiator shall save the data pointers as soon as the last byte of the last iuCRC for a SPI information unit is transferred. The save shall occur even if the initiator detects an error in the SPI data information unit. If a target retries an operation it shall send a MODIFY DATA POINTER~~S~~ or MODIFY BIDIRECTIONAL ~~READ~~ DATA POINTERS message, then request that the SPI data information unit be transferred again.

Section 14.3.2 Table 41 – Type
04h Data
Sent by a target to indicate a SPI data information unit shall follow this SPI L_Q information unit. The DATA LENGTH field shall not be set to zero. For a bidirectional command, the SPI data information unit shall be sent from the initiator to the target if the BIDIRECTIONAL DIRECTION field is set to 01b or shall be sent from the target to the initiator if the BIDIRECTIONAL DIRECTION field is set to 10b.

05h Data Stream
Sent by a target to indicate an unspecified number of SPI data stream information units shall follow this SPI L_Q information unit. The DATA LENGTH field shall not be set to zero. For a bidirectional command, the SPI data stream information unit(s) shall be sent from the initiator to the target if the BIDIRECTIONAL DIRECTION field is set to 01b or shall be sent from the target to the initiator if the BIDIRECTIONAL DIRECTION field is set to 10b.

~~[Add two more types:]~~
~~06h Bidirectional Read Data~~
~~Sent by a target to indicate a SPI data information unit shall follow this SPI L_Q information unit. The DATA LENGTH field shall not be set to zero. For a bidirectional command, the SPI data information unit shall be sent from the target to the initiator.~~

~~07h Bidirectional Read Data Stream~~
~~Sent by a target to indicate an unspecified number of SPI data stream information units shall follow this SPI L_Q information unit. The DATA LENGTH field shall not be set to zero. For a bidirectional command, the SPI data stream information unit(s) shall be sent from the target to the initiator.~~

[Add this new field to table 40 SPI L_Q Information Unit]
[add this text below the table:]
The BIDIRECTIONAL DIRECTION field (byte 16 bits 7:6) determines the data direction if the command is a bidirectional command and the type code is Data or Data Stream. The field shall be set to 00b for a unidirectional command or a type code other than Data or Data Stream, 01b for a bidirectional command transferring data from the initiator to the target, or 10b for a bidirectional command transferring data from the target to the initiator. 11b is reserved.

15 SCSI pointers
The initiator provides for a set of ~~three~~ pointers for each task, called the saved pointers. The set of ~~three~~ pointers consist of one for the command, one (for unidirectional commands) or two (for bidirectional commands) for the data (data-out and data-in), and one for the status. When a send command service is received from an application client, the task’s ~~three~~ saved pointers are copied into the initiator's set of ~~three~~ active pointers. There is only one set of active pointers in each initiator. The active pointers point to the next command, data-out, data-in, or status byte to be transferred between the initiator and the target. The saved and active pointers reside in the initiator.

The saved command pointer always points to the start of the command descriptor block for the task. The saved status pointer always points to the start of the status area for the task. The saved data-out pointer points to the start of the data-out area until the target sends a SAVE DATA POINTERS message for the task or after the initiator successfully ~~receives or~~ transmits a SPI data information unit. The saved data-in pointer points to the start of the data-in area until the target sends a SAVE DATA POINTERS message for the task or after the initiator successfully receives a SPI data information unit.

In response to the SAVE DATA POINTERS message or successful receipt or transmission of a SPI data information unit, the initiator stores the values of the current data-out and data-in pointers into the saved data-out and data-in ~~-~~pointers for that task. The target may restore the active pointers to the saved pointer values for the current task by sending a RESTORE POINTERS message to the initiator. The initiator then copies the set of saved pointers into the set of active pointers. Whenever a target does a physical disconnect from the bus, only the set of saved pointers are retained. The set of active pointers is restored from the set of saved pointers upon a physical reconnection of the task or a successful receipt of a SPI L_Q information unit.

Since the data pointer values may be modified by the target before the task ends, ~~it~~ they should not be used to test for actual transfer length because the value may no longer be valid.

[added in rev 3:]
If information units are disabled, the target shall do a physical disconnect and physical reconnect to change the data transfer direction.

Figures 63, 64, 65, Section 14.3.2, and Section 16.3.2
Change “data pointer” to “data pointers”.

Section 16.3.8 MODIFY DATA POINTER
The MODIFY DATA POINTER message (see table 54) is sent from the target to the initiator and requests that the signed ARGUMENT be added (two's complement) to the value of the current data pointer. ~~In a unidirectional command, t~~The data pointer is whichever of the data-out or data-in pointers is being used by the command. ~~In a bidirectional command, the data pointer is the data-out pointer.~~ The ~~E~~enable ~~M~~modify ~~D~~data ~~P~~pointer (EMDP) bit in the disconnect-reconnect mode page (see 18.1.2) indicates whether or not the target is permitted to issue the MODIFY DATA POINTER message. The target shall only issue the MODIFY DATA POINTER message during a unidirectional command.

It is recommended that the target not attempt to move the data pointer outside the range addressed by the command. Initiators may or may not place further restrictions on the acceptable values. Should the target send an ARGUMENT value that is not supported by the initiator, the initiator may reject the value by responding with the MESSAGE REJECT message. In this case, the data pointer is not changed from its value prior to the rejected MODIFY DATA POINTER message.

Section 16.3.8 MODIFY BIDIRECTIONAL ~~READ~~ DATA POINTERS
The MODIFY BIDIRECTIONAL ~~DATA~~ POINTERS message (see table ~~54~~xx) is sent from the target to the initiator and requests that the signed DATA-OUT ARGUMENT be added (two's complement) to the value of the current data-out~~in~~ pointer and the signed DATA-IN ARGUMENT be added (two’s complement) to the value of the current data-in pointer. The enable modify data pointer (EMDP) bit in the disconnect-reconnect mode page (see 18.1.2) indicates whether or not the target is permitted to issue the MODIFY BIDIRECTIONAL ~~READ~~ DATA POINTERS message. The target shall only issue the MODIFY BIDIRECTIONAL ~~READ~~ DATA POINTERS message during a bidirectional command.

~~It is recommended that the target not attempt to move the data-in pointer outside the range addressed by the command. Initiators may or may not place further restrictions on the acceptable values. Should the target send an ARGUMENT value that is not supported by the initiator, the initiator may reject the value by responding with the MESSAGE REJECT message. In this case, the data-in pointer is not changed from its value prior to the rejected MODIFY BIDIRECTIONAL READ DATA POINTER message.~~

[by putting both values in one message, error handling becomes more complicated. I assume that if an error occurs in either pointer, both of the changes are rejected.]

It is recommended that the target not attempt to move the data-out pointer or data-in pointer outside the range addressed by the command. Initiators may or may not place further restrictions on the acceptable values. Should the target send a DATA-OUT ARGUMENT or DATA-IN ARGUMENT value that is not supported by the initiator, the initiator may reject the values by responding with the MESSAGE REJECT message. In this case, both the data-out pointer and data-in pointer are not changed from their values prior to the rejected MODIFY BIDIRECTIONAL DATA POINTERS message.

[assign a new extended message code for this message, either 02h or 05h. This appears in tables 49 and 50.]
~~[duplicate the MODIFY DATA POINTER table here, with the new extended message code and the name renamed to MODIFY BIDIRECTIONAL READ DATA POINTER]~~
[add this new table:]

Table xx. MODIFY BIDIRECTIONAL DATA POINTERS message format.

	[Bit numbers]
0	Extended Message (01h)
1	Extended Message Length (09h)
2	Modify Bidirectional Data Pointers (TBD 02h or 05h)
3	(MSB)
4	Data-Out Argument
5	
6	(LSB)
7	(MSB)
8	Data-In Argument
9	
A (or 10)	(LSB)

Section 16.3.12 RESTORE POINTERS
The RESTORE POINTERS message is sent from a target to direct the initiator to copy the most recently saved command, data, and status pointers for the task to the corresponding active pointers. The command and status pointers shall be restored to the beginning of the present command and status areas. The data pointers shall be restored to either the values at the beginning of the data areas in the absence of a SAVE DATA POINTERS message or to the values at the point at which the last SAVE DATA POINTERS message occurred for that task.

When information unit transfers are enabled there are implied restore pointers. For more information on this see 14.1 and 14.3.3.

Section 16.3.13 SAVE DATA POINTERS
The SAVE DATA POINTERS message is sent from a target to direct the initiator to copy the current data pointers to the saved data pointers for the current task.

Section 18.1.2 Disconnect-reconnect mode page
The enable modify data pointer (EMDP) bit indicates whether or not the initiator allows the MODIFY DATA POINTER and MODIFY BIDIRECTIONAL ~~READ~~ DATA POINTERS messages to be issued by the target. If the EMDP bit is a zero, the target shall not issue the MODIFY DATA POINTER and MODIFY BIDIRECTIONAL ~~READ~~ DATA POINTERS messages. If the EMDP bit is a one, the target is allowed to issue MODIFY DATA POINTER and MODIFY BIDIRECTIONAL ~~READ~~ DATA POINTERS messages.

If the EMDP bit is a one and the initiator responds to a MODIFY DATA POINTER or MODIFY BIDIRECTIONAL ~~READ~~ DATA POINTERS message with a MESSAGE REJECT, then the target shall return a CHECK CONDITION. The sense key shall be set to ABORTED COMMAND and the sense code shall be set to INVALID MESSAGE ERROR.

Section 19.3.1 Application client SCSI command services overview
The SCSI command services shall be requested by the application client using a procedure call defined as:
Service response = execute command (
target SCSI ID+logical unit number[+tag],
command descriptor block,
[task attribute],
[link control function],
[data-in buffer size],
[data-out buffer],
[~~command byte count~~data-out buffer size] ~~||| [data-in buffer]~~,
[data-in buffer],
status,
service response).

Section 19.4.1 Device server SCSI command services overview
The SCSI data buffer movement services shall be requested from the device server using a procedure call defined as:
Service response = move data buffer (
target SCSI ID+initiator SCSI ID+logical unit number [+tag],
device server buffer,
application client buffer offset,
request byte count ||).

~~Only one type of data buffer movement procedure call shall be used while processing one command, either data-in delivery or data-out delivery.~~ Either data-in delivery, data-out delivery, both data-in and data-out delivery, or neither data delivery may be used while processing one command. If both are used (for a bidirectional command), the device server shall combine the data-in and data-out service responses into one service response.