To: X3T9.2 Committee Members   X3T9.2/93-098r0

Date: 18 May 93 13:48 GMT

From: "Smyers, Scott" <SMYERS.S@applelink.apple.com>

Subject: SBP WG mtg 5/6 - 5/7/93

Serial Bus Protocol Working Group
Santa Clara, CA
May 6 and 7, 1993

These minutes document the Serial Bus Protocol working group meeting held at Apple's Monroe facility on May 6 and 7, 1993.

Attendance

Gerald Marazas, IBM/Boca Raton
Stephen Scandalis, Texas Instruments
Larry Lamers, Maxtor
Jeff Stai, Western Digital
Ralph Gee, IBM/Adstar
Greg Floryance, IBM/Adstar
Nathan Hays, Quantum
Scott Smyers, Apple Computer

Detailed review of comments received from Greg Floryance

Greg started out by expressing that he would like to have the meetings start on time.

In preparation for this meeting, Greg had gone through the SBP document and prepared 5 handouts containing notes and other suggested input. Greg began by going through his document which we identified as 5/5/93; GGF1. This document described some conclusions that he, Jerry and Scott came to regarding editorial and wording changes to the SBP document.

The group heard each of Greg's points from the GGF1 document and accepted each of them, with the following exceptions:

> For Isochronous control register and isochronous command FIFO, there is not a requirement that the target implement one of these for each stream. The target returns the address of these 2 at login time and there is not requirement that the target return unique addresses for each stream. This is implementation dependent.
>
> The name "command FIFO" remains (Greg had a preference for changing the name to "target FIFO")

During the discussions, Nate proposed that we add a definition for isochronous commands and asynchronous commands and spell out how they're delivered. After some mulling around, we came up with the following:

> Add a statement that all commands (asynchronous and isochronous) are delivered using the same mechanism
>
> Differentiate isochronous and asynchronous commands in the definitions by spelling out that they differ in how the data is transferred in response to these commands
>
> Rename the command block to Command Data Structure (CDS) and define the CDS in the definitions section

Also during this discussion we agreed to come back to the question of adding definitions for isochronous fetch policy and asynchronous fetch policy and thereby help the reader understand the differences and the lack of interaction between the fetching of isochronous chains and the fetching of asynchronous chains.

We next moved on to Greg’s document 5/5/93; GGF2. The first item was removed.

On the next point, we identified that devices can implement either asynchronous data transfer only, isochronous data transfer only or both. We then fell into a discussion of if your a device is isochronous only, should we be able to deliver an INQUIRY command to the isochronous command FIFO? There was quite a bit of discussion here.

Eventually, we agreed that we would add a pointer to an isochronous login FIFO in the configuration ROM. We also agreed to restrict the use of the login FIFO’s to CDS’s defined in the SBP standards document. The SBP document will note that for the CDS’s that contain a field for holding a CDB, that CDB is defined in the SCSI-2 document or one of the SCSI-3 command set documents only (no new CDB’s are defined in the SBP document).

The following is the proposed configuration ROM information for an SBP target:

Length	CRC
01[key]	pointer to SBP async login FIFO
01[key]	pointer to SBP isoch login FIFO

Note that the 6 bit key value in the upper byte of each of these entries identifies the meaning of the low order 24 bits. Therefore, either one of these pointers can exist without the other, and the order of these pointers relative to each other is not important.

If a target configuration ROM contains a pointer to an async login FIFO, then that target implements async commands. If a target configuration ROM contains a pointer to an isoch login FIFO, then that target implements isoch commands. Note that the target configuration ROM can contain either of these individually, or both without regard to order.

We next spent some time creating a table which identifies the function of the asynchronous login FIFO and the isochronous login FIFO

Async login FIFO	Isoch login FIFO
async Log in operation	isoch Log in operation
async CDS fetch policy	isoch CDS fetch policy
SAM task set	SAM task set (???)
SBP tap mgmt commands	no tap slot management
optional AE sign in	no sign in
async xfer of status block	async xfer of status block
async xfer of CDS’s	async xfer of CDS’s
async xfer of media data	isoch transfer of media data

There was a suggestion that we remove the notion of ACA for isochronous data transfer, and specify that you cannot send ACA commands to the isochronous command FIFO. With some further discussion, we came up with the following set of rules:

> It is an error to send an ACA command to any FIFO unless an ACA condition is true
>
> When an ACA condition is true, ACA commands can only be sent to the ACA

FIFO

ACA commands shall not be sent to the isochronous command FIFO

Addressing this point in the standard document, Scott had prepared some proposed text. With some amendments, this text was accepted for the standard.

Continuing again with GGF2, Greg's comments were accepted as written, though some points drew some discussion.

Greg presented his 5/5/93; GGF3 which is a proposal for inclusion into the SBP document. His submission includes a diagram of the command fetch algorithms and some accompanying text. Eventually, however, we got onto the subject of terminology regarding tap versus fetch versus reading a command.

With a whole lot more discussion, we eventually arrived at the identification of 2 boundaries and 3 possible locations, or states, for a CDS:

1. CDS is in initiator address space and the target has no knowledge of it

2. CDS has been transported to the target (either as the contents of a tap message, or as the result of a target read operation)

3. CDS is entered into the target's task set, in which the SAM rules of command reordering apply

We also decided on some terms for the action of commands going from one state to another:

A CDS is "acquired" when the command is transferred from the initiator to the target, either as the result of a command tap or a target read operation

A CDS is "entered" into the task set

We then talked about isochronous data during which Scott described the changes to isochronous in response to input received at the last meeting:

a new command called the streamControl command, which performs the same function as the old ICR

Some things were taken out of the streamSetup command

ICR changed to an isochronous control FIFO, which receives only streamControl commands

New open issues, including parameterizing a drive's capabilities, worst case latency for a control operation

The group discussed these issues for a time, then decided to sleep on it and discuss them the next day.

As the last point for the day, Jerry brought up 4 points of discussion:

Allocation of tap slots - it was decided to leave it as documented in SBP revision 10i. The target is not required to enforce its allocation of tap slots. The mechanism is only suitable for a way for multiple initiators to cooperatively manage the limited tap slot resources.

Minimum number of tap slots - it was decided to put this requirement into the SCSI-3 device dependent profile documents, but not in SBP

Location of Next Command Pointer in CDS - The Next Command Address field of the CDS was left where it is at the beginning of the CDS

Return of number of free tap slots in status block - It was decided that Jerry and Scott will document a proposal for including information about available tap slots in the status block.

On the morning of the second day, the group spent some time coming up with a diagram which describes the flow of CDS's in the target. It was generally decided that the diagram should be incorporated into the SBP document.

The group next discussed the issue of isochronous data and SBP. Larry Lamers took the following notes on the isochronous discussions:

Notes from Larry Lamers on Friday's discussions

acquire: the act of transferring a CDS from an initiator to a target. [This can be done with a fetch (read) or a tap (write).]

CDScommand data structure. The data structure used to communicate a command to a target.

enqueue:

enter: the act of enqueuing a SCSI-3 command into task set.

fetch: A target read of a CDS.

enter policy: The rules governing the enqueueing of commands.

tap: An initiator write of the first CDS of a chain.

chain queue: a set of elements that are pointers to the chain of CDS structures in host memory.

Synch period ?

cycle #25/20

channel # 8

continure/event/error/action/byteoffset4/4/4/4/16

Nate wants the above to be in CDS's instead of CSR's.

The group felt that an isoch control channel should be used for flow control/pause/resume functions.

Change stream control CDS to

```
   stream ID
      channel number  with new name of data channel number
      control channel number
      continue mode
      error reporting
```

Isochronous control packet:

```
      start/stop/pause control
      sy field value to be sent on next cycle
```

Scott wants the sy field recorded on the disk as "sub-channel" information. A count-key-data method that returns a mirror image of the data & control streams is possible.

Scott vetoed using a control channel; he proposed a CSR quadlet for control and accepted the fact that there would be several milliseconds latency.