SONY
Network & Software Technology Center of America
24034 NE 29th Street
Sammamish, Washington 98074-5468
(425) 558-0464
(425) 558-0575 (fax)

FROM: John Nels Fuller

TO: T10 SBP-3 Working Group

DATE: October 10, 2001

SUBJECT: Additional requirements for SBP-3 devices

Upon a review of draft SBP3r01e I discovered that some things relating to the new features of the CSR Architecture (IEEE P1212) and to bridging (IEEE P1394.1) are missing from the draft. I propose the following changes be incorporated into a subsequent draft to remedy this situation.

Unit directories can now appear in either the root directory or in instance directories. We want to discourage them in root directories so that migration to the use of instance directories can begin. I hope the following text reflects this:

7.3.3 Unit_Directory entry

The Unit_Directory entry is a directory entry in the root directory or an instance directory that describes the location of a unit directory within configuration ROM. A Unit_Directory entry should only appear in the root directory when necessary for compatibility with legacy (SBP-2) initiators. There may be more than one unit directory; each unit directory shall be located by a separate Unit_Directory entry. It is strongly recommended that all unit directories be referenced by Unit_Directory entries in instance directories. It is permissible for a particular unit directory to be referenced by Unit_Directory entries in both the root directory and in an instance directory. Figure 52 shows the format of this entry.

We want to require the use of keywords so that devices looking for certain types of functions can quickly move on to a more interesting node and so that DEP may be used for discovery across bridges. The following new section is my attempt to make sure that the appropriate keywords are included in both the master keyword leaf and the other keyword leaves, for which I have invented the term “instance keyword leaf” (I don’t know if we should add a section to explain the format of a keyword leaf or just refer to P1212, for now I’ve opted for the latter):

7.3.4 Keyword_Leaf entry

The Keyword_Leaf entry is a directory entry in the root directory and instance directories that describes the location of a keyword leaf within configuration ROM. The root directory of an SBP-3 target shall contain a single Keyword_Leaf entry; this refers to the master keyword leaf. An instance directory that contains one or more Unit_Directory entries for SBP-3 units shall also contain a single Keyword_Leaf entry; this refers to an instance keyword leaf. Keyword leaves are described in IEEE P1212. The instance keyword leaf for an instance directory that references one or more SBP-3 unit directories shall contain the keyword “SBP” in addition to any keywords required by any command sets implemented by those SBP-3 units and any other keywords deemed appropriate by the implementer. The master keyword leaf for an SBP-3 target shall contain the keyword “SBP” in addition to any keywords required by any command sets implemented by any SBP-3 units and any other keywords deemed appropriate by the implementer. Figure *xx* shows the format of this entry.

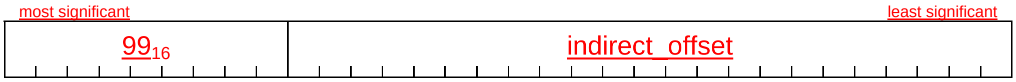

Figure *xx* – Keyword_Leaf entry format

99_{16} is the concatenation of *key_type* and *key_value* for the Keyword_Leaf entry.

The *indirect_offset* field specifies the number of quadlets from the address of the Keyword_Leaf entry to the address of the master keyword leaf within configuration ROM.

We want to encourage the use of instance directories so we had better mention them and give a description of how they are used and why you might want to make them hierarchical. I have chosen not to try to duplicate what is in P1212 but just to summarize it for the reader and refer her to that standard for details:

7.3.5 Instance_Directory entry

The Instance_Directory entry is a directory entry in the root directory or an instance directory that describes the location of an instance directory within configuration ROM. An instance directory describes a single function of a device and contains entries such as Feature_Directory entries, Keyword_Leaf entries, Unit_Directory entries, and Instance_Directory entries. When an instance directory contains multiple Unit_Directory entries these represent alternate methods of accessing the same device function. When an instance directory contains Instance_Directory entries then those child instance directories each represent a different portion of the function described by the parent instance directory. For example, a DVD-ROM changer could be represented as a parent instance directory containing a Unit_Directory entry referencing a unit that controls both the medium changer and data access features of the device in the same unit while one child instance directory would provide a Unit_Directory entry that controls only the medium changer and another child instance directory would provide a Unit_Directory entry that controls only the data access to the medium. Instance directories are described in IEEE P1212. Figure *xx* shows the format of this entry.

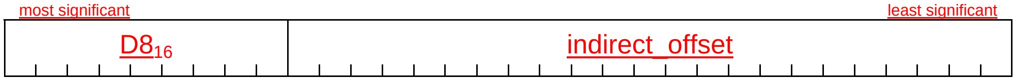

Figure *xx* – Instance_Directory entry format

$D8_{16}$ is the concatenation of *key_type* and *key_value* for the Instance_Directory entry.

The *indirect_offset* field specifies the number of quadlets from the address of the Instance_Directory entry to the address of the instance directory within configuration ROM.

We must have a discovery mechanism for the bridged environment and DEP is the only contender at the moment. In order for DEP to work we need to make sure that every SBP-3 device will respond to the DEP requests and so we will require both initiator and target to do so. However, we do not want to tie our hands if a better solution comes along, so we will merely recommend that initiators and targets use DEP requests to find each other. Here’s the text:

A.4 Discovery and Enumeration Protocol

Both initiators and targets shall respond to DEP requests as specified in IEEE P1394.1, Annex D. Both initiators and targets should use DEP to discover the global node ID of remote nodes with which they must communicate.

There is an advantage to using bridge aware logins even if all the nodes involved are on the local bus, namely that the connection persists through local bus resets. This means that devices must reestablish who each node on the bus is as quickly as possible, hopefully without contributing to the flurry of traffic that happens after a local bus reset. So we want to encourage the use of bus topology analysis. Here's my attempt:

A.5 Bus Topology Analysis

Since the login connection between initiator and target does not require a reconnect after a local bus reset when the original login request was for bridge aware mode it is necessary for both the initiator and target to reestablish their mappings of node identities (EUI-64) to local physical IDs after a local bus reset. In order to minimize the amount of local traffic on the bus during the critical period after a local bus reset it is strongly recommended that both initiators and targets implement the bus topology analysis algorithm described in IEEE P1394.1, Annex E.