Test Methodology for Characterizing the SEE Response of a Commercial IEEE 1394 Serial Bus (FireWire)

C. Seidleck (Raytheon, Lanham MD), S. Buchner (QSS, Landover MD),
H. Kim (Jackson & Tull, Washington DC), P.W. Marshall (Consultant, Brookreal VA),
and K. LaBel (NASA GSFC, Greenbelt MD)

Abstract - ***The SEE response of an IEEE 1394 FireWire serial bus was tested with heavy ions and protons. A unique approach to testing and categorizing the Single-Event Effects (SEEs) is presented.***

I. INTRODUCTION

NUMEROUS sensors on board future spacecraft will generate vast amounts of data for processing and storage. This will require that the data be moved within the spacecraft from the sensors to processors and memories across a bus with sufficient bandwidth to handle all the data.

One bus being considered for future space applications is the IEEE 1394 FireWire digital serial bus. This bus is a commercial-off-the-shelf (COTS) device that offers superior performance. However, being a COTS device, the IEEE 1394 is likely to be quite sensitive to SEEs caused by the ionizing radiation in space. Therefore, a study of the SEE response of an IEEE 1394 serial bus from Texas Instruments (TI) was undertaken using heavy ions.

Two types of errors were observed during SEE testing: "soft errors" that did not disrupt data transmission across the bus, and "hard errors" that did. "Hard errors," termed single-event functional interrupts (SEFIs), were of particular interest because of the multitude of failure modes manifested. Nine different types of SEFIs were observed and classified according to what steps were required to restore proper bus operation. Some of the SEFIs required restarting the software for transmission to resume. Others required the cable be removed and then reinserted. The most severe cases involved a "cold reboot" of the entire system. The measured SEFI LET threshold of below 4 MeV.cm^2/mg is sufficiently low to render this part in its current form unsuitable for use in a space radiation environment.

II. BACKGROUND

IEEE 1394 is a formal description of the architecture of FireWire, an advanced digital serial bus used for transmitting data between devices, and originally developed by Apple Computer. FireWire is relatively inexpensive, frequency scalable, able to transmit in two modes (asynchronous and isochronous), and permits "plug-and-play" operation. The cable version is unique in that power is distributed through the cable for operation of the transceiver's repeating function even if the node power is off.

IEEE 1394 describes the hardware and software necessary for communications via three protocol levels – transaction layer, link layer and physical layer. The functions of these protocol layers can best be understood by describing their roles during data transfer across the bus from *requester*, or initiator, of the transfer to *responder*, or recipient, of the data.

As mentioned above, IEEE 1394 supports both isochronous and asynchronous transfer modes. Which transfer mode is used depends on the nature of the data to be transmitted. The salient features of each transfer mode will first be described.

The isochronous mode is used for transmitting signals that require constant data transfer rates, such as audio and video broadcasts, and for which error-free data is not critical. Isochronous transfers can occur between a requester and any number of responders and there is normally no acknowledgement that the targeted node has received the data. Bit errors will have a minor impact on, for example, video images transferred in real time from a camera to a video monitor.

Asynchronous transfers are more complicated and are employed in those cases where data transfer must be error-free and where a constant transfer rate is not required, such as for data files. Therefore, asynchronous transfers are not guaranteed a fixed amount of bandwidth. They occur between a requester node and a targeted responder node with a unique address. The successful transfer of valid data onto the bus and capture by the targeted application is relayed back to the initiator. Should the data be corrupted in any way, the transfer initiator is requested to resend the data.

Figure 1 shows the topology of the transaction, link and physical layers. Only the link and physical layers are involved in isochronous data transfers. The link

layer, which provides the interface between the isochronous software driver and the physical layer, codes data packets for sending across the cable and decodes any packets received across the cable. The link layer first determines whether the received data was meant for that node, and, if so, what software driver it should be forwarded to. The physical layer provides the actual interface between the link layer and the cable.

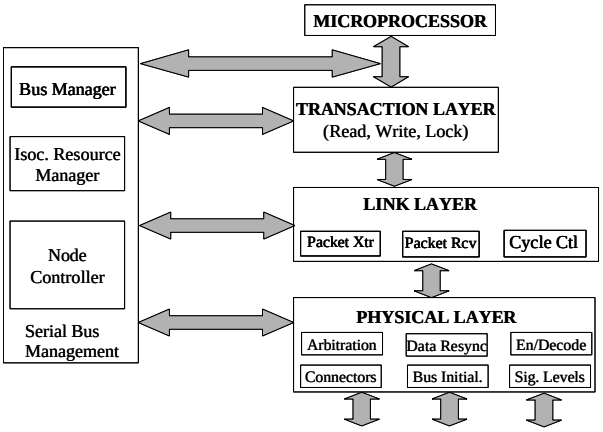

Figure 1. Protocol layers used in the IEEE 1394 implementation.

Asynchronous data transfers involve the transaction layer as well as the link and physical layers. The transaction layer supports specific functions, such as read, write and lock, related to asynchronous transfers. The requestor transaction layer initiates the request to send data and notifies the responder of the request. The responder transaction layer then returns status or data to the requester and notifies the requester that the response has been received. The link layer, which provides the interface between the transaction and physical layers, also creates and decodes data packets. As in the case of isochronous transfers, the physical layer provides the actual interface between the link layer and the cable.

For both asynchronous and isochronous communications, the data is transferred in the form of packets consisting of a header section followed by the data section. The packets are assembled in shared computer memory and not on the IEEE 1394 board. Because the packet header and data are in separate memory locations, the header contains an address field pointing to the data. Just prior to transmission, the header and data are combined to form a complete packet.

Figure 2 shows an example of the data packet used for asynchronous data transmission. The first 150 bits constitute the header and contain information required for routing the data, such as the identities of the source and destination, what kind of transaction is being used, cyclic redundancy codes (CRCs) to warn of corrupted

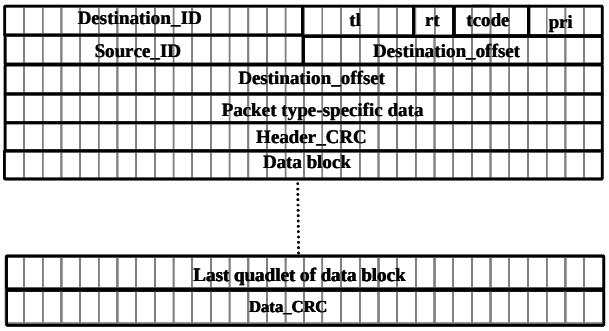

Figure 2. Asynchronous packet format. (Destination_ID = bus address and physical ID of the destination node, tl = transaction identifier sent by requester for this transaction, rt = retry identification and protocol specification, tcode = defines packet format, pri = priority, Source_ID = identity of node sending packet, Destination_offset = address location within the target node, Header_CRC = CRC value for the header, Datablock = bits containing data to be transmitter, Data_CRC = CRC value for the data.)

data, etc. The isochronous stream packet has similar header and data sections, but the formats differ slightly from the asynchronous packet format.

The proper functioning of the bus is controlled by information stored in registers on both the physical and link layer chips on the IEEE 1394 board. The link chip contains 102 Open Host Controller Interface (OHCI) registers and 22 Peripheral Component Interconnect (PCI) registers. In addition, the link contains a First-In-First-Out (FIFO) register through which the packet passes on its way from memory to the bus. Thus, the FIFO only contains important data when the packet passes through. The physical layer chip contains 16 internal registers.

There are two registers on the link that are of particular importance for asynchronous data transmission. They are the “Context Command Pointer Register” (CCPR) and the “Context Control Register.”(CCR) There are two similar registers for isochronous transfers. The CCPR registers point to the address in computer memory of the header. The CCR registers contain a “run” bit, which, when set by

software, causes the packet pointed to by the CCPR to be sent to the bus.

In summary, data packets are assembled in computer memory and then, upon instructions from the CCPR and CCR registers, passed through a FIFO to a bus. Registers on the physical and link layer chips control the operation of the bus. Irradiating only the 1394 board with heavy ions confines single-event upsets (SEUs) to the OHCI, PCI and FIFO registers on the link layer chip and to the internal registers on the physical layer chip. This paper reports on how those SEUs affect communications in the IEEE 1394 digital serial bus – from "soft" errors that have no effect on communications to SEFIs that disrupt communications. A unique aspect of this study was the development of a method for categorizing SEFIs according to what steps were required to restart communications.

III. Parts Selected

Table I lists the part identification numbers including the transaction/link layer and physical layer chips together with the development board. For heavy-ion testing, the plastic covering the chips was etched away so that the ions could reach the sensitive regions of both the link and the physical layer chips.

TABLE I.
PART NUMBERS FOR THE IEEE 1394 FIREWIRE FROM TI.

Link Part #	Phy. Part #	Devel. Board
TSB12LV26PZT	TSB41AB3PFP	TSBKOHCI403

IV. Testing Procedure

For SEE testing, communications were established between a control (CTRL) computer and a device-under-test (DUT) computer. Figure 3 shows the test configuration. Both computers contained an IEEE 1394 board and were linked together with an IEEE 1394 cable. The DUT computer was located in the accelerator test vault with its IEEE 1394 board plugged into the PCI backplane, which was removed from the computer case and firmly secured to the top of the case to facilitate ion beam irradiation of the physical and link chips separately. A bus extender was used for the DUT computer's monitor, keyboard and mouse so that a single operator could conveniently control both computers from one location outside the accelerator testing area. In addition, a PCI bus isolation card was used together with a digital multimeter for monitoring board current.

Software directed the CNTRL computer to send a message to the DUT computer instructing it to poll the registers in the link and send that information back to the CNTRL computer. The CNTRL computer then compared the data received from the DUT computer with what it expected. Any differences were flagged as "soft errors" and stored in memory. The same sequence was repeated until transmission was interrupted, either by user intervention or by a SEFI.

Testing was done for both asynchronous and isochronous modes. For both modes, the contents of 42 of the 102 OHCI and 21 of the 22 PCI registers in the DUT computer's Link chip were polled. The remaining registers were too volatile to monitor, as were all the registers in the physical layer chip.

The request packets assembled in the CNTRL computer's memory consisted of a header with the information required for communicating with the DUT and a data block containing the instructions directing the DUT to poll the registers contained on the DUT's link chip. The DUT computer's response packets differed

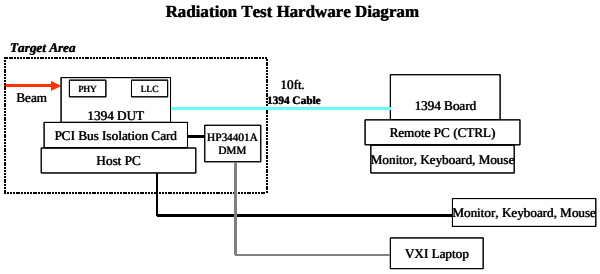

Figure. 4. Experimental setup showing the two computers, one inside the test vault and the other outside and connected with a 1394 cable.

slightly in that the header contained information required for communicating with the CNTRL computer, and the data block contained data obtained from polling the 21 PCI and 42 OCHI registers on the DUT's link chip.

After communications were established between the CNTRL and DUT computers, the DUT's 1394 board was exposed to a series of heavy ions with different LETs. "Soft" errors were continuously logged and the board current was monitored until the ion beam was halted following a SEFI or a predetermined fluence. During testing it became clear that there were many different kinds of SEFIs and that one possible way of classifying them was by the steps taken to restart communications. It should be mentioned that rebooting the DUT or both the DUT and CNTRL computers was very time-consuming, requiring up to five minutes to restore communications. This affected the amount of data collected because of the costs of accelerator time.

V. Results

A. *Soft Errors (SEUs)*

Three kinds of soft errors were detected: i) errors that self-corrected and for which the board current did not increase, ii) errors that self-corrected but for which the board current increased from 18 mA to 44 mA, and iii) increases in board current from 18 mA to 44 mA

with no errors. Table II shows the types of soft errors observed in the link chip when running in both asynchronous and isochronous modes.

TABLE II.
TYPES OF SOFT ERRORS OBSERVED IN LINK

	Asynch.	Isoch.
Error, I:18mA->44mA	X	X
Error, $\Delta I = 0$	X	
No error, I: 18mA -> 44 mA	X	X

Figure 1 shows the cross-section for soft errors as a function of ion LET for both isochronous and asynchronous modes when the link was irradiated. The LET threshold for soft errors in the link running both asynchronous and isochronous modes was below 3 MeV.cm^2/mg. These SEUs occurred only in the 21 PCI and 42 OHCI registers monitored as well as in the FIFO. Because the registers on the physical chip were not monitored, it was not possible to construct a curve of SEU cross-section as a function of ion effective LET.

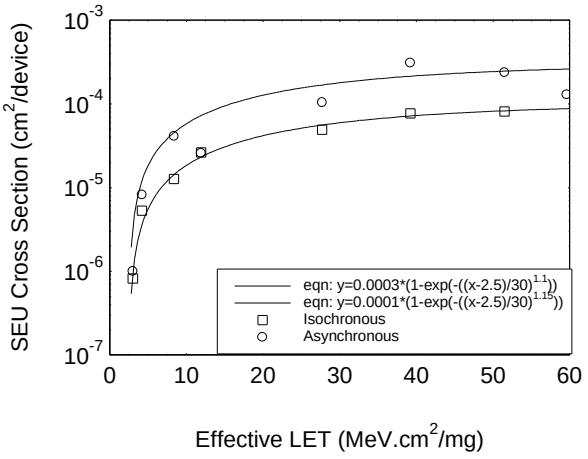

Figure 5. Cross-section per device for "soft" errors as a function of ion effective LET for the link. Included are Weibull fits to the data.

B. Hard Errors (SEFIs)

By definition, SEFIs interrupted communications and required some form of intervention to restart. SEFIs occurred during irradiation of the physical and link chips, running either asynchronous or isochronous modes. Table III shows how the SEFIs were categorized according to the steps required for restarting communications. The steps are listed from software restarts to hardware reboots. The first step simply involved restarting the software loop on the CNTRL computer. The second step involved resetting the bus followed by starting the software loop. The third step consisted of reloading the software application, resetting the bus and starting the test loop. An interesting type of SEFI (labeled number 4) was one it was possible to verify that the CNTRL computer sent a register data solicit request to the DUT computer. It was also possible to verify that the DUT received the register data solicit request, and that it sent the data packet back to the CNTRL computer. However, the CNTRL computer could not read the register data response packet, which meant that communications were interrupted and the CNTRL computer had to be "cold" rebooted. Another interesting SEFI, labeled number 5, required the removal and reinsertion of the cable to restart communications. Because IEEE 1394 permits plug-and-play, removing the cable removes power from the board, and reinserting it causes a complete reconfiguration of the system,

TABLE III.
STEPS TAKEN TO RECOVER FROM SEFIS.

Step	Action
1	SEU test loop restarted on CNTRL i.e. packet sent to DUT requesting register information.
2	Software bus reset, i.e., force CTRL to be root, initiate bus reset in the PHY, and reset Link to restore OHCI registers and flush FIFOs.
3	Reload Software application, which refreshes lockdown memory region shared by the software and hardware. Implies steps 2,1.
4	Controller cannot see the register data response packet. Power cycle the CNTRL followed by steps 3,2,1.
5	Disconnect/reconnect 1394 cable. Causes hard bus reset and tree ID process.
6	Step 5 followed by steps 3, 2, and 1.
7	Step 5 followed by cold rebooting only DUT followed by steps 3, 2, and 1.
8	Cold reboot DUT followed by steps 3, 2 and 1.
9	Reboot both CNTRL and DUT PCs, followed by steps 3, 2, and 1.

assigning node identities, root etc. Steps 6 and 7 also involved removing and reinserting the cable, but communications would only resume after additional steps (listed in Table III) were taken. Some SEFIs required "cold" rebooting of the DUT computer alone, whereas others required "cold" rebooting both.

Table IV lists the type of SEFI observed for the link and physical chips running asynchronous or isochronous modes.

TABLE IV
TYPES OF SEFIS OBSERVED

Step	Link (Asynchr)	Link (Isochr)	Physical (Asynchr)	Physical (Isochr)

1	X	X	X	
2		X	X	X
3	X	X		
4				X
5			X	X
6	X	X	X	X
7	X	X		
8	X	X		X
9	X	X	X	X

For analysis, all the SEFIs resulting from irradiation of the link chip were combined, as were all of the SEFIs originating in the physical chip. Figure 6 shows the SEFI cross-section as a function of ion effective LET for the link chip. Figure 7 shows the SEFI cross-section as a function of ion effective LET for the physical chip.

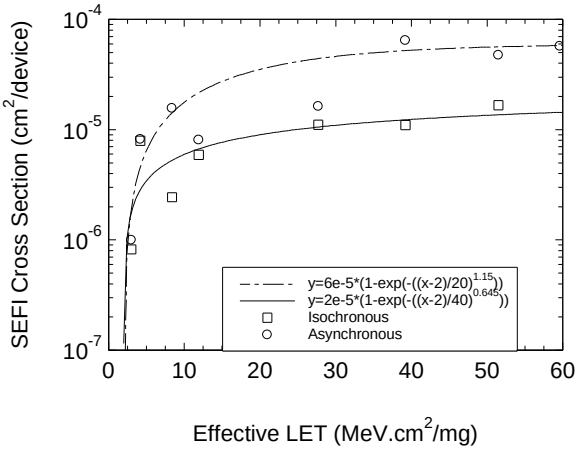

Figure 6. SEFI cross-section per device as a function of ion effective LET for the link chip.

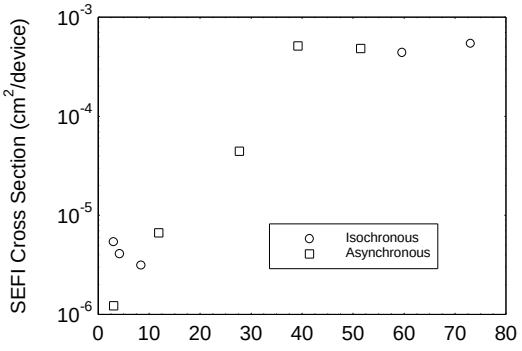

Figure 7. SEFI cross-section per device as a function of ion effective LET for the physical chip.

VI. Discussion

A. *Soft Errors (SEUs)*

As previously mentioned soft errors were the result of SEUs in registers on OCHI, PCI and FIFO registers on the link chip. Not all of the registers could be monitored for SEUs because the data they contained was extremely volatile. Only 42 of the OHCI registers and 21 of the PCI registers were monitored. The remaining registers on the link chip as well as all the registers on the physical chip were not checked for SEUs. This does not imply that SEUs did not occur in those registers or that they had no effect, only that they were not recorded as soft errors.

The presence of a SEU was detected when the registers were polled following the arrival of the data solicit packet. The data response packet sent by the DUT computer to the CNTRL computer contained the erroneous bit, which was recognized as a "soft" error by the CNTRL computer, and recorded in a file. After recording the SEU, the registers on the DUT computer were scrubbed of any remaining errors and normal communications resumed.

The soft errors recorded during the radiation exposure of the link chip had no effect on the operation of the bus. One can therefore conclude that they occurred in data not used by the bus in the particular application used in this radiation test. In a different application, such as would be the case if more nodes were communicating with each other, some of the SEUs that were harmless in this application may result in SEFIs. This stresses the importance of doing SEE testing under the same conditions as would be used in an actual space application.

An example of a soft error is one in the 32-bit Asynchronous Request Filter Low (ARFL) Register on the link chip during asynchronous transmission. The function of the ARFL register is to enable reception of asynchronous request packets. When an asynchronous request packet is received, the source node ID is examined. If the bit corresponding to the node ID is not set in this register, the packet is not acknowledged and the request is not queued. In this example, the register is configured such that only asynchronous request packets from nodes 0 and 1 will be accepted. This is implemented by setting the two least significant bits to "1". All the other bits in the register are set to "0." Therefore, if a SEU switches a different bit (bit 26) from a "0" to a "1" asynchronous request packets from node 26 will be accepted. However, since there is no node 26, the result is a SEU that has no effect on data transmission.

Some errors were accompanied by an increase in the board current whereas others were not. The origin of the current increase is not known at the present time. It could have been due to a mini-latch that had no effect on communications, or it could be due to a change in the operating conditions that did not manifest itself in any other way. Similarly, in some cases increases in current were observed but no errors were recorded. Those increases in current could have been associated with

SEUs in some of the registers that were not monitored because of their volatility.

Some SEUs originating in the FIFO were also detected. This attribution was the result of a careful examination of the data packets transmitted from the DUT computer to the CNTRL computer. The data are transmitted in quadlets (32 bits) but not all the bits in the quadlet are required for the data. The unused bits are set to zero to complete the quadlet. A SEU in one of the unused bits could obviously not have come from a register and must instead have originated in the FIFO through which the transmitted packet passed on its way to the bus. Clearly, a SEU that occurred in the data part of the quadlet while the quadlet was passing through the FIFO could not be distinguished from one that occurred when the data was still in the register.

Figure 5 shows that the LET threshold for SEUs was below 3 MeV.cm^2/mg, which is lower than the LET threshold for SEFIs (3 - 4.2 MeV.cm^2/mg). Evidently, the more SEU sensitive registers are those that do not result in SEFIs. Also the SEU cross-section for the asynchronous mode exceeds that for the isochronous mode, a not surprising result given the more complicated nature of asynchronous transmissions.

B. SEFIs

SEFIs are serious failure modes because significant intervention is necessary to restart communications. Table IV shows that there are nine different kinds of SEFIs, classified according to what steps were required to restart communications. The mildest intervention consisted of resending a data solicit packet to reset the register with a SEU. This was sufficient to restart normal bus operations. The most drastic recovery from a SEFI consisted of rebooting both computers, a process that also required the most time and is, therefore, the least desirable form of SEFI.

It is worth noting that for ions with low LETs (2-10 MeV.cm^2/mg) the most common SEFIs were those for which recovery was relatively simple consisting of steps 1, 2 or 3. In contrast, ions with higher LETs (> 10 MeV.cm^2/mg) produced a greater proportion of SEFIs that necessitated rebooting either the DUT computer by itself or both the DUT and the CNTRL computers. There is no obvious reason why the more complex recoveries are associated with higher LET ions and the simpler ones with low LET ions.

The table shows that there was very little difference between the types of SEFIs measured for the isochronous and asynchronous modes when the link chip was irradiated. However, the relative sensitivities do differ. The fact that no SEFIs involving recovery mode 2 were observed could mean that others had higher cross-sections and effectively masked it, or that the associated registers were simply not sensitive.

Table IV suggests that the types of SEFIs that occur when the physical chip is irradiated depend on the transmission node. For instance, rebooting the DUT alone to recover from a SEFI in the asynchronous mode was never necessary, whereas just resending a data solicit packet was found never to be effective in the isochronous mode.

The Table also shows that some SEFIs (#3 and #7) originate only in the link chip, whereas others originate only in the physical chip (#4 and #5). Explanations for these phenomena require a more detailed study, possibly involving injecting errors into known locations on the chips using an ion microprobe or a pulsed laser beam.

Figure 6 shows the SEFI cross-section as a function of ion effective LET for the link. All the different kinds of SEFIs are counted as one type because the lack of statistics on each type would give very poor quality curves. Treating all the SEFIs as one results in a relatively smooth curve that can easily be fit with a Weibull function for use in SEFI rate predictions. As for the case of SEUs, the SEFI cross-section is greater when running in asynchronous than in isochronous mode. The very low LET threshold for SEFIs (below 4 MeV.cm^2/mg) suggests that this implementation of the IEEE 1394 bus is not suitable for space applications.

Figure 7 shows the SEFI cross-section as a function of ion effective LET for the physical chip. As for Figure 6, all the SEFIs are treated as one, but the statistics are not as good. Therefore, no attempt was made to fit a Weibull function to the data. The data in the figure indicate that the cross-sections for the physical chip running asynchronous and isochronous modes are comparable over the entire LET range.

As an example of a SEFI we again consider the ARFL register in which the two least significant bits are each set to “1” as required when only two nodes are communicating with each other. Should a SEU occur in one of these bits, switching it from a “1” to a “0,” the reception of asynchronous request packets would be denied and a SEFI would result. As more nodes are added to the bus, more of the bits would be set to “1” and the SEFI cross-section would increase.

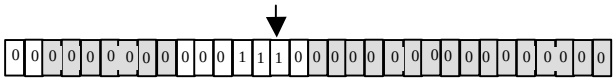

Figure 8. Contents of the Host Controller Control Register on the link chip. The bits highlighted in grey are reserved and so cannot be changed by software.

Figure 8 shows the contents of the Host Controller Control Register (HCCR) on the link chip. Another example of how a SEFI can occur would be if a SEU occurred in bit 17 (indicated by the arrow) on the HCCR. This bit is set to “1” when the system is ready to begin operation. If an upset switched it to a “0,” the link would immediately be disconnected from the 1394 bus.

No packets would be received or transmitted and communications would be halted between the CNTRL and DUT computers. To resume communications, a "1" would have to be restored via software instruction.

VII. Conclusions

From the results of SEE testing discussed above, it is clear that performing a full SEE characterization of the IEEE 1394 is a complicated and time consuming task. The cross-section at each ion LET must be measured for four different cases, consisting of the link and physical chips running in asynchronous and isochronous modes.

The results also confirm the obvious point that testing should be done under the same conditions as will apply in space. Therefore, if more than two nodes will be communicating with one another, the testing should be done with more than two nodes connected together.

Finally, the fact that the SEFIs could be categorized into nine unique recovery modes of differing complexity does not imply that all nine recovery modes should be included in a space application. Disconnecting and reconnecting the cable is clearly not a practical solution for a space application. Rebooting the entire system following any SEFI is a drastic but apparently necessary solution.

VIII. References