111167 N95- 17582

31987

SPACECRAFT DATA SIMULATOR FOR THE TEST OF LEVEL ZERO PROCESSING SYSTEMS

p.10

Jeff Shi, Julie Gordon
RMS Technologies, Inc.
Code 520.9

Chandru Mirchandani, Diem Nguyen
Loral AeroSys
Code 521

NASA, Goddard Space Flight Center
Greenbelt, Maryland 20771

ABSTRACT

The Microelectronic Systems Branch (MSB) at Goddard Space Flight Center (GSFC) has developed a Spacecraft Data Simulator (SDS) to support the development, test, and verification of prototype and production Level Zero Processing (LZP) systems. Based on a disk array system, the SDS is capable of generating large test data sets up to 5 Gigabytes and outputting serial test data at rates up to 80 Mbps. The SDS supports data formats including NASA Communication (Nascom) blocks, Consultative Committee for Space Data Systems (CCSDS) Version 1 & 2 frames and packets, and all the Advanced Orbiting Systems (AOS) services. The capability to simulate both sequential and non-sequential time-ordered downlink data streams with errors and gaps is crucial to test LZP systems. This paper describes the system architecture, hardware and software designs, and test data designs. Examples of test data designs are included to illustrate the application of the SDS.

1. INTRODUCTION

Simulation of spacecraft data is a basic function of any space and ground data system. Over the years, many simulation systems have been developed to support spacecraft and ground system Integration and Test (I&T). However, very few are capable of testing LZP systems that are based on CCSDS recommended data formats [1][2] and require large unique test data sets with complex data scenarios.

In 1992, the Microelectronic Systems Branch (MSB) at GSFC developed a Spacecraft Data Simulator (SDS) to support it's Very Large Scale Integration (VLSI) LZP system prototype phase one (VLSI LZP-1) development [3]. The VLSI LZP-1 is capable of performing LZP functions for CCSDS packet telemetry at rates up to 20 Megabits per second (Mbps). In order to test this system, it was necessary to simulate realistic streams of spacecraft data, including a variety of errors and data gaps. The SDS was used to simulate data streams of 750 Mbytes each (i.e., 5-minute sessions at 20 Mbps) with all valid timecodes and sequence counts. To emulate onboard tape recorders, reversed data streams were generated for playback sessions. To test the overlap deletion function, the forward real-time sessions and reversed playback sessions were simulated with regions of overlap at the beginning and end of sessions. Many types of errors were inserted into selected frames and packets in both overlap and non-overlap regions to fully exercise system capability of handling errors.

The SDS was significantly enhanced through the development and deployment of Fast Auroral Snapshot Explorer (FAST) Packet Processing System (PPS), that is based on the VLSI LZP-1 architecture [4]. The use of Solid-State Recorders (SSR) onboard FAST created many complicated data scenarios never before encountered in conventional tape recorder-based missions [5]. New functions were added to the SDS to simulate the scenarios, such as, downloading thousands of data fragments in a single session, interleaving real-time and playback data with identical packets being present on different Virtual Channels (VC), overlap occurring anywhere in a data stream, and packet sampling that resulted in non-contiguous packet sequence counts. These enhancements enabled the SDS to be used successfully in the FAST PPS system development, integration, and acceptance testing.

Early in 1994, the SDS was upgraded further to support the development of VLSI LZP prototype phase two (VLSI LZP-2) [6]. Simulation software was modified to support CCSDS AOS data formats, and key hardware components were upgraded increasing the maximum data output rate from 25 to 80 Mbps.

2. SYSTEM OVERVIEW

The Spacecraft Data Simulator consists of a hardware subsystem and a software subsystem. The hardware subsystem provides processing power, data storage, a network interface, and a telemetry interface. The software subsystem takes in user specifications and generates simulated CCSDS telemetry data accordingly.

The hardware subsystem, shown in Figure 1, contains a simulation processor and a 5.5 Gbytes disk farm. The simulation processor is contained in a Versa Module Eurocard (VME) equipment rack; it consists of a Master Controller card, an Ethernet Interface card, a Disk Controller card, a Memory card, and a MSB-developed Data Generator (DG) card. All cards except the DG card are Commercial Off-the-Shelf (COTS) components that provide system base functions for data processing, buffering, and network interfacing. The DG interfaces with the disk farm and is used to store simulated telemetry data in the disk farm during offline data generation. During a test session, the DG retrieves the data from the disk farm, serializes it and outputs it through an RS-422 interface or an Emitter Coupled Logic (ECL) interface. The data rate is user-selectable from 0 to 80 MHz. The disk farm parallel architecture enables data transfer at rates up to 128 Mbps. The disk farm capacity is configurable from 5.5 to 40 Gbytes.

The software subsystem consists of two major components: the Test Pattern Generator (TPGEN) and the Large Volume Data Generator (LVGEN). TPGEN is a menu-driven software package that can generate small sets of Nascom blocks, and conventional and AOS CCSDS frames and packets for up to 32 VCs and 256 Application Processes (AP). Packet and frame placement is user-defined. The data can be either Cyclic Redundancy Checked (CRC) or Reed-Solomon encoded, and many types of errors can be inserted in the data. TPGEN has been used to support many missions including the Topographic Explorer (TOPEX); Solar Anomalous and Magnetospheric Particle Explorer (SAMPEX); and FAST.

LVGEN is a script-based package. It uses TPGEN to configure "base sets" of frames or blocks when the ratio and placement of packets from all sources are defined. It then switches among the "base sets" while generating the test data set, with all timecodes and sequence counts increment correctly during switches. The only restriction is that all "base sets" must share the same source list.

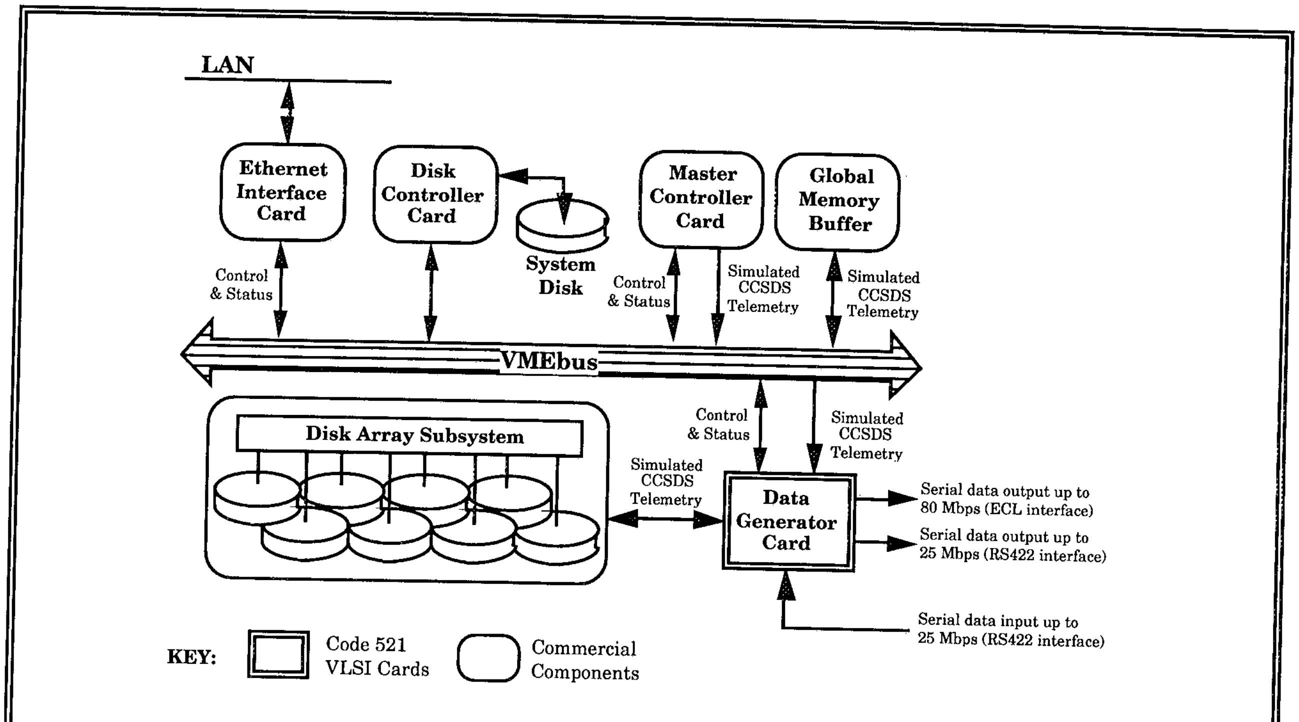
LAN
Ethernet Interface Card
Disk Controller Card
System Disk
Master Controller Card
Global Memory Buffer
Control & Status
Control & Status
Simulated CCSDS Telemetry
Simulated CCSDS Telemetry
VMEbus
Disk Array Subsystem
Control & Status
Simulated CCSDS Telemetry
Simulated CCSDS Telemetry
Data Generator Card
Serial data output up to 80 Mbps (ECL interface)
Serial data output up to 25 Mbps (RS422 interface)
Serial data input up to 25 Mbps (RS422 interface)
KEY:
Code 521 VLSI Cards
Commercial Components

Figure 1. System Block Diagram

Figure 2 illustrates the information and process flow in SDS data set generation. Spacecraft needs are interpreted as requirements and operational scenarios. Requirements are used to design the data set and to specify the errors to be injected. The data set design includes simulating the expected order, frequency, and content of the data from the spacecraft. This step produces external files defining the complexity of the data, scripts encompassing the order and rules for error injection and data generation, and the TPGEN catalogs (configuration files) defining the repeatable sets of frames. LVGEN uses the products from the data design and error injection stage to generate the data. The user interface downloads the test setup, and activates the DG to output the simulated data stream using the LVGEN-generated data.

When a "base set" must contain a larger number of frames than can be defined in a single TPGEN configuration file, it is broken down into "subsets," each with a separate configuration file. Different versions of the same "subset" may be used to simulate the forward playback scenario where identical packets from one source can be transmitted on more than one VC. The start count for each file created by LVGEN defines the regions of overlap between the sessions.

For "complex data," where the transmission of packets out of time order by a spacecraft must be simulated, LVGEN fills the sequence count and timecodes for each packet from external files that are generated by a Packet Header Definition (PHD) utility based on the test scenarios. The simulated science or engineering source data can be simple repeated patterns, or can be read from files that may contain real spacecraft data, still images, or other interesting data patterns.

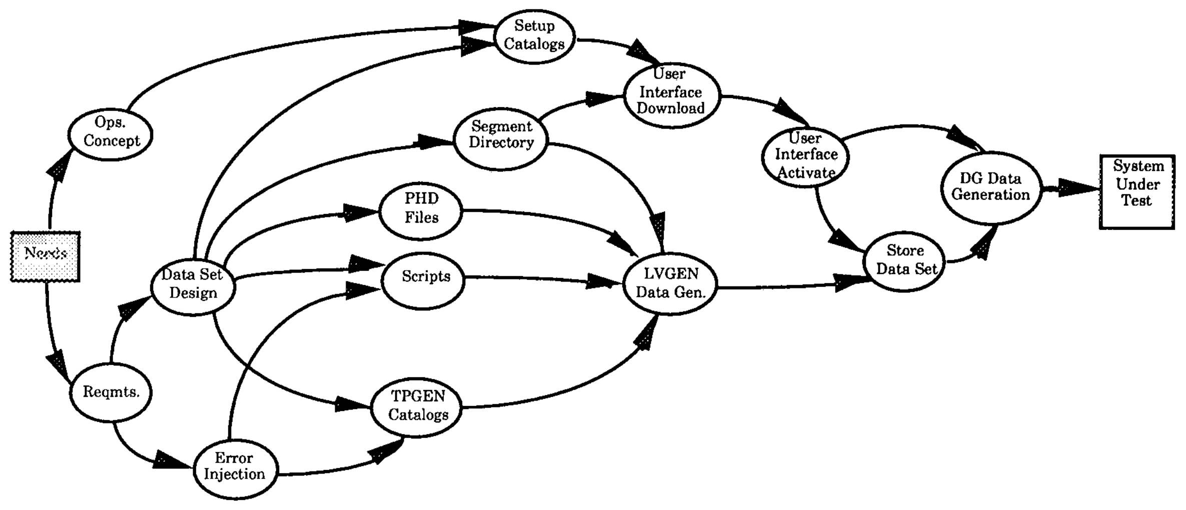

Figure 2. Data Set Generation Flow Diagram

3. SDS APPLICATION

The SDS was used to functionally test the FAST PPS system through system development, integration, and acceptance testing. This section will describe the strategy used to test the FAST PPS which included exercising the functionality and performance of the PPS. The strategy was to initially develop the minimum number of data sets which would base-line the functionality of the FAST PPS; these data sets will have varying degrees of complexity. The next step was to place errors in the initially-generated data sets that would cover the range of errors and performance metrics to fully stress the PPS.

3.1 CALIBRATING DATA SET GENERATION

The first data set that was generated was referred to as 'clean' or 'calibrating,' i.e., with no errors. It consisted of a single session of block-encoded frame data, with 564,131 blocks containing 300,600 telemetry frames that simulated a 30-minute session at 1.5 Mbps.

The calibrating data set simulated the expected proportions of the VCs as closely as possible. Within each VC, relative proportions of Application Process Identifiers (APID) were configured as specified in the FAST Data Management Plan. In order to expedite simulation, frames of the same VC occurred in groups of not more than 1280 frames within the telemetry stream. The data set consisted of playback and realtime interleaved data, bound on both sides by VC 7 fill data (see Figure 3).

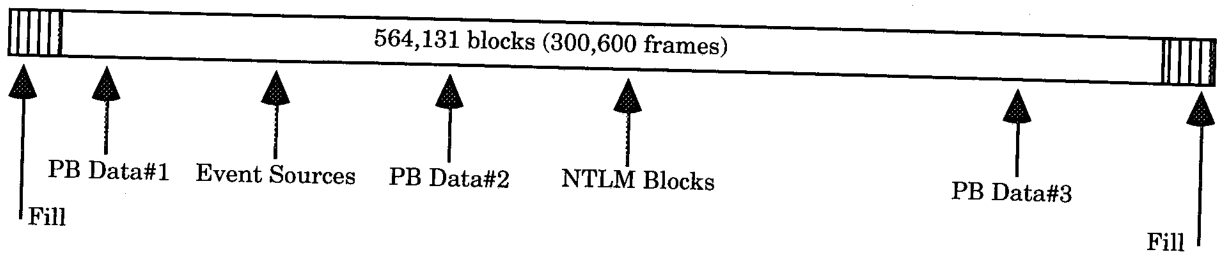

Figure 3. The Calibrating Data Set Composition

The playback data, VC 1, was dumped three times during the session at a rate of 1 out of every 5 frames downloaded. During the time VC 1 data was output, VC 2, 3, and 4 data continued to be read out in a similar order as when no VC 1 data was being dumped, although the contents of each VC were slightly different due to simulation limitations. The first VC 1 dump occurred at the beginning of the session; the second dump occurred in the middle of the session; and the last dump was a partial dump that occurred near the end of the session. VC 0 was output at a rate of 1 frame every 8 seconds (approximately 1 frame every 1394 frames at 1.5 Mbps). VCs 2, 3, and 4 were distributed throughout the scenario in alternating groups of each VC. For this data set, it was decided that VCs 2, 3 and 4 would have relative densities 8:0.125:1. One APID had all its packets in VC 3. Several other APIDs had packets in both VC 3 and 4; these APIDs were in VC 4 most of the time but switched to having small numbers of packets in VC 3 no more than 20 times during the course of the session.

A group of "non-telemetry" blocks was introduced in the middle of the data stream, to test telemetry/non-telemetry filtering. These blocks had different destination codes or message type codes.

The calibrating data set was generated by LVGEN using TPGEN catalogs, PHD files and user developed scripts. Variances in the data content were produced by switching between base sets while generating the data. Although the data was Nascom block encoded, the base sets were defined by the number of frames and the content of the frame data per base set.

3.2 DATA SET COMPLEXITY

Different types of complex data containing multiple segments were simulated to fully test the capability of the FAST PPS. "Segment" is a concept referring to a group of packets of the same source which has a consistent time order, either forward or reversed. Data is "simple" if there is only a single segment received per session per source; data is "complex" if there are multiple segments received from a source during a session. For the FAST spacecraft, there are three ways in which complex data may be transmitted; each of these data types were simulated in the calibrating data set.

For all sources in VC 2, the FAST spacecraft uses a "sampling" storage algorithm to write data into the onboard SSR. Sensor (source) data is compared against a preset threshold. If the data value is less than the threshold, the data is filled sequentially into a small buffer storage area on the SSR. If the data value meets or exceeds the preset threshold, a sample of the subsequent sensor data is sequentially stored in a partition within a larger buffer storage area on the SSR. After the larger buffer storage area is full, samples are still taken and compared to all the stored samples according to pre-defined criteria. If a new sample is better, it will overwrite an old sample with the best value, that may be anywhere from the 1st partition to the Nth partition. As more observations are made and the sensor value fluctuates, this scheme will result in segmented data in the SSR. When the SSR is dumped sequentially from low to high address, the PPS will receive and reassemble data segments that are completely out of time order.

In LVGEN, the FAST sampling process was simulated by placing the first packet of the defined "sample" at the start of the second segment; and the remaining packets were then used to fill the second, the first, and the third segments, in that order. This means, among other things, that if the playback list, defined by external files, used by LVGEN specified adjacent forward-time-order samples, the last packet of the third segment of the first sample was continuous with the first packet of the first segment of the second sample. The proportion of the first segment to the second segment for each sample was controllable within LVGEN. In the calibrating data set, VC 2 contained "complex data," with 8 simulated "samples" received in a mixed time order.

The second source of multiple segments in FAST data was the VC 1 engineering playback data. During a pass, the FAST spacecraft may re-transmit stored engineering data several times, resulting in several wholly or partially identical groups of packets which must be overlap processed to delete redundant data. Multiple transmissions of playback engineering data and the mixture of realtime and playback data (albeit on separate VCs) was also simulated in the calibrating data set. The third type of FAST data segments result from the splitting of survey science data between VC 3 and 4. On the FAST PPS, these two streams, which contain no overlapping data, will be processed separately and merged, similar to the way VC 0 and 1 data were processed.

The calibrating data set was a sequential data set, using a sequence count increment of 1. Some complexity was introduced to base-line the functionality of the FAST PPS. In this data set, only some VC 2 APIDs were fully simulated as sampled. The remaining sources were "simple," i.e., a single segment per session. The order of groups of packets within the data stream was specified by means of listing the playback order of samples for VC 2 sources in the calibrating data set. Two additional calibrating data sets were designed with greater complexity and used to further stress the capability of the FAST PPS. They were also used to test non-sequential packet data processing.

3.3 DATA SET ERRORS AND THEIR IMPLEMENTATION

The SDS was also used to simulate data streams with errors. Three scenarios were needed to fully test the FAST PPS: (1) Introduce various types of errors, (2) Introduce a large number of errors, 5% and 10%, and (3) Introduce errors in sequential and non-sequential data, both partially and completely overlapped, to exercise the capability to segment and locate questionable, or 'fuzzy' packets.

The first scenario introduced errors in the calibrating data set to simulate realistic cause and effects of data and transmission errors. The errors that can be generated include: (1) Noise in the transmission from space-to-ground, and (2) Noise in the ground-to-GSFC transmission. The errors that can be generated, and the resultant effects at block, frame, and packet levels are shown in Table 1.

Table 1. Errors, Results, and Statistics

ERRORS	RESULTS	STATISTICS
Block Synchronization Pattern Errors	Dropped Blocks, Missing Frames, Missing Packets	Number of Blocks, Number of Block Synchronization Errors, Number of Block Sequence Errors, Number of Frames, Number of [Missing, Back to Search, Lock, Check] Frames, Number of Packets, Number of Missing Packets, Number of Packet Gaps
Block & Frame CRC Errors	Tagged Blocks, Bad Frames, Bad Packets	Number of Block Polynomial Errors, Number of Frame CRC Errors, Number of Packet Errors
Block CRC Errors	Tagged Blocks, Good Frames, Good Packets	Number of Block Polynomial Errors, Number of Frames from CRC Error Blocks, Number of Packets from CRC Error Blocks
Frame Synchronization Errors	Missing Frames, Missing Packets	Number of Frames, Number of [Missing, Back to Search, Lock, Check] Frames, Number of Packets, Number of Missing Packets, Number of Packet Gaps
Frame First Header Pointer Errors	Rejected Frames, Missing Packets	Number of FHP Errors, Number of Missing Frames, Number of Missing Packets, Number of Packet Gaps
Frame CRC Errors	Bad Frames, Bad Packets	Number of Frame CRC Errors, Number of Packet Errors
Frame Bit Slip Errors	Bad Frames, Good Packets	Number of Bit Slips, Number of Missing Frames, Number of Missing Packets, Number of Packet Gaps
Packet Length Errors	Bad Packets	Number of Missing Packets, Number of Packet Length Errors, Number of Packet Sequence Errors

To introduce these errors, TPGEN catalogs for the calibrating data set were modified to include the errors. To ensure the number of times an error is to be repeated and its location in blocks, frames, and packets, base sets were split into smaller sets and the subsequent modified catalogs were used in conjunction with LVGEN to generate data with induced errors. For example, to generate an error in the 10th frame of a 40 frame base set within a specific repetition of the base set, (i.e., frame location,) it was necessary to break the base set into two smaller base sets of 9 and 31 frames, respectively. The sequence of the frames is still maintained, but to inject the error, the LVGEN script will treat these two smaller base sets separately. For example, a data set requires repeating a particular base set (BS3) 200 times, within which the 102nd repetition will have some error introduced in the 10th frame. BS3 is split into BS3a and BS3b, and the script will specify repeating BS3 101 times, repeat BS3a once, BS3b once, and subsequently repeat BS3 98 times.

Three types of simulated block errors were adequate to test the requirements and statistics at block, frame, and packet levels; Block Synchronization Error - by a flipped or incorrect bit in the block synchronization pattern; Block Polynomial Error with Frame CRC Error - by an error of one or more bits in the Nascom Block; and Block Polynomial Error with No Frame CRC Error - by flipped or incorrect errors in the Nascom block header or trailer.

Four types of simulated frame errors were adequate to test the requirements and test statistics at frame and packet levels; Frame Synchronization Error - flipped or incorrect bit in the frame synchronization pattern; Frame CRC Error without Block Polynomial Error - by one or more incorrect bit in the telemetry data field; Frame Bit Slip Error - by dropped or extra bit(s) in the frame; and Frame First Header Pointer Error - error in the first header field of the frame.

Packet length error was simulated to test the system requirements for detection and qualification of packet level errors. It was simulated by flipping the last significant bit in the packet length field specified in the packet header of the selected packet.

The next scenario tested the performance of the system and evaluated its capability to process data with large quantities of errors. Two data sets with 'High Volume Errors' were designed to test these capabilities. They consisted of the FAST VC 2 sources only, with frames and packets similar to previous data sets, to simplify generation and predicted results. High-rate sampling was not simulated; all sources were a single segment per session. Frame CRC, First Header Pointer Errors, Frame Synchronization, Block Synchronization, and Packet Length errors were distributed throughout the data set to bring the total percentage of packets with errors or gaps to 5% and 10%, respectively.

The final scenario tested the capability of the system to process the maximum number of sources with non-sequential and questionable packet sequence numbers. A data set called SRC200 was designed to test the requirement that the FAST PPS was able to process up to 200 sources. It was not intended to otherwise simulate the expected FAST telemetry format, neither in APIDs nor in packet sizes. The opportunity presented by this non-realistic scenario allowed the SRC200 data set to contain several "complex" sources with a full range of segment processing tests, including non-sequential and sequential overlap cases. This data set had no induced errors or gaps, but was used as the template for a later test set in which the complex data was used to test segment processing with fuzzy packets and non-sequential segment processing with errors and gaps. The 200 sources were generated by distributing 128 APIDs over VCs 0-6. The frame and packet format for SRC200 was the same as for previous data sets. Packet size was specified as 1054 bytes for all packets, resulting in one packet per frame for all sources.

4. CURRENT AND FUTURE DEVELOPMENT

During 1994, the Spacecraft Data Simulator was upgraded to support development and I&T of VLSI LZP prototype phase 2 that performs LZP functions at rates up to 50 Mbps. The Data Generator card was outfitted with a new mezzanine that increases output data rates from 25 to 80 Mbps. The TPGEN and LVGEN data generation packages have been modified to simulate CCSDS AOS data for up to 32 VCs and 256 APIDs. To test the system's ability to provide AOS services, the SDS simulated AOS Coded Virtual Channel Data Units (CVCDU) consisting of Private Data Units for Virtual Channel Access Service, Insert Zone Data Units for Insert Service, Bitstream Data Units for Bitstream Service, fixed and variable length packets within Multiplexed Packet Data Units (MPDUs) with repeated APIDs on different VCs. The CVCDUs contained incomplete and fill packets, and included the full range of errors as described in the FAST PPS test scenarios.

In the current implementation, all error injection scenarios and the foibles introduced in the data sets are designed in advance and manually catalogued through the TPGEN menu before the data generation process is initiated. As data scenarios get more and more complicated, this can become a tedious and time consuming process. Each time a scenario changes, even just adding or removing one error, the entire data set has to be regenerated.

MSB is currently engaging in the development of a second generation data simulation package. Based on UNIX platform, this package will be highly modular and script-driven, and can be ported to different UNIX workstations. CCSDS telemetry data will be simulated by first generating a set of basic data units, then piping it through a series of simulation modules, each of which is responsible for one layer of CCSDS protocol or one type of data manipulation. Data or error characteristics will be specified through scripts. Graphical-based user interfaces will be provided to help users design, generate, modify, and examine their test data sets. While maintaining all SDS capabilities as described in this paper, this layered and modularized architecture will greatly improve efficiency.

5. SUMMARY

SDS has demonstrated its versatility and flexibility by supporting the LZP project through all phases of development. Its unique capabilities to simulate realistic spacecraft CCSDS data streams, especially SSR data scenarios, proved to be invaluable for system I&T of the VLSI LZP prototype phase 1 and 2 systems as well as the FAST PPS. The knowledge and expertise gained in the development of the current SDS will be used to develop the new generation of data simulators capable of testing systems running at speeds in excess of 300 Mbps.

6. REFERENCES

1. "Packet Telemetry," CCSDS 102.0-B-3, Blue Book, Consultative Committee for Space Data Systems, November, 1992.
2. "Advanced Orbiting Systems, Networks, And Data Links," CCSDS 701.0-B-2, Blue Book, Consultative Committee for Space Data Systems, November, 1992.
3. Shi, J., Horner, W., Grebowsky, G., Chesney, J., "A Prototype VLSI Level Zero Processing System Utilizing the Functional Component Approach," Proceedings of International Telemetering Conference, 1991, pp. 519-531.
4. Shi, J., Horner, W., Grebowsky, G., Chesney, J., "Fast Auroral Snapshot Explorer (FAST) Packet Processing System (PPS)," Proceedings of International Telemetering Conference, 1993, pp. 445-459.
5. Shi, J., Mao, T., Clotworthy, T., Grebowsky, G., "Lessons Learned Supporting On-Board Solid-State Recorders," Space Ops 94.
6. Shi, J., Harris, J., Speciale, N., Bennett, T., "A Second Generation 50 Mbps VLSI Level Zero Processing System Prototype," Space Ops 94.

7. NOMENCLATURE

AOS	Advanced Orbiting Systems
AP	Application Process
APID	Application Process Identifier
CCSDS	Consultative Committee for Space Data Systems
COTS	Commercial Off-the-Shelf
CRC	Cyclic Redundancy Check
CVCDU	Coded Virtual Channel Data Units
DG	Data Generator
ECL	Emitter Coupled Logic
FAST	Fast Auroral Snapshot Explorer
GSFC	Goddard Space Flight Center
I&T	Integration and Test
LVGEN	Large Volume Data Generator
LZP	Level Zero Processing
MPDU	Multiplexed Packet Data Units
MSB	Microelectronic Systems Branch
Nascom	NASA Communications
PED	Polynomial Error Detector
PHD	Packet Header Definition
PPS	Packet Processing System
SAMPEX	Solar Anomalous and Magnetospheric Particle Explorer
SDS	Spacecraft Data Simulator
SSR	Solid-State Recorder
TOPEX	Topographical Explorer
TPGEN	Test Pattern Generator
VC	Virtual Channel
VME	Versa Module Eurocard
VLSI	Very Large Scale Integration

Systems Engineering

* *Presented in Poster Session*