

## **Implementation of a Modern Internet Protocol-Based Communications System and Error Detection and Correction System for Commercial Memory within a Radiation Hardened FPGA for a Low-Earth-Orbit Satellite**

Dillon M. Collins, Brendan S. Surrusco, Sven G. Bilén, Charles L. Croskey  
Communications and Space Sciences Laboratory, Electrical Engineering Department  
*The Pennsylvania State University, University Park, PA, 16802*

Anthony Jordan and Ron Lake  
*Aeroflex Colorado Springs, Inc.*

**ABSTRACT:** There is a growing interest in the application of common terrestrial communications protocols, such as the Internet Protocol (IP), to spacecraft systems. There is also a desire to increase computing power through the use of commercial products with unknown or limited radiation survivability. Such interests are driven by the need to reduce the high costs of space technology while both standardizing and increasing capability. As part of the research process for Penn State's Local Ionospheric Measurements Satellite (LionSat), a communications encoder was developed that implements IP and bi-phase L (BPL) encoding measures along with a data synchronizer to allow operation of a synchronous downlink using asynchronous data from the flight computer. Also developed was an error-detection-and-correction scheme to protect commercial high-speed SDRAM from multiple independent bit errors for a 32-bit Linux system. Both of these designs were created for implementation in a RadHard Aeroflex Eclipse FPGA and require a combined total of less than 45% of available logic cells. The resulting system provides a compact radiation-tolerant solution for the communication and memory assurance needs of a modern satellite flight computer.

### **NOMENCLATURE**

*kbps* = kilo-bit per second  
*ksp/s* = kilo-symbol per second

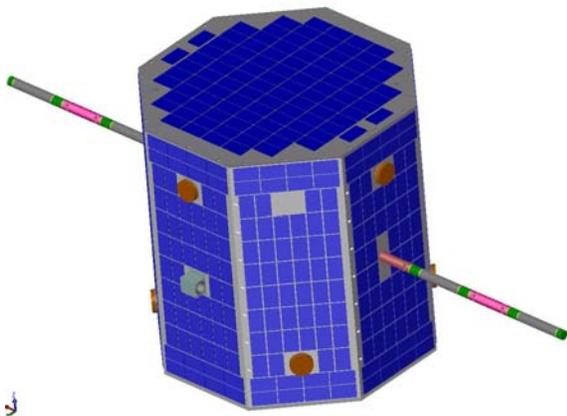
### **INTRODUCTION**

Over the last few years, the spacecraft industry has been growing tremendously as both commercial and government interests in space have greatly expanded. Compared to a few decades ago, the heavens are now filled with satellites for commercial and military communications, TV, broadband Internet access, and scientific missions. Missions to other objects in our solar system have also increased dramatically, yet the capabilities of qualified spacecraft hardware still lag significantly behind the demand. When compared to the commercial technology sector, satellite systems come up incredibly short in cost vs. capability comparisons. Not surprisingly, spacecraft engineers are turning to commercial industry to find answers to their technology resource needs for significantly less cost than traditional sources. Some believe that the very future of space systems depends heavily on the

ability of engineers to take advantage of the inexpensive, yet ever increasing, computing power and plug-and-play resources available to the average consumer. However, as those who have already begun to use commercial devices in space systems have found, many consumer products do not perform well in the space radiation environment.

In the spirit of education and furthering the integration of commercial technology into the spacecraft industry, The Pennsylvania State University has been developing a new LEO nanosatellite with systems composed almost entirely of commercial parts running a commercial operating system. The Local Ionospheric Measurements Satellite (LionSat) project was started with sponsorship from the American Institute of Aeronautics and Astronautics (AIAA), the National Aeronautics and Space Administration Goddard Space Flight Center (NASA GSFC), the Air Force

Office of Scientific Research (AFOSR), and the Air Force Research Laboratory Space Vehicles Directorate (AFRL/VS) as part of the University Nanosat-3 (NS-3) program. LionSat, shown in Figure 1, is an octagonal nanosatellite measuring 18.5 inches in length by 18.25 inches in diameter. The primary mission for LionSat is to study the plasma environment surrounding the satellite using a pair of newly developed hybrid plasma probes and to test a micro-RF ion thruster design under development at Penn State.<sup>1,2</sup> As a secondary mission, LionSat will be used to demonstrate the application of the IP (Internet Protocol) to a space-ground link.<sup>3,4,5</sup>



**Figure 1. LionSat model with solar cell patterns and plasma probes depicted.**

## SYSTEM REQUIREMENTS

For some time now, the Internet Protocol has been gaining attention as a useful tool in satellite applications. As part of the LionSat mission, the engineers decided to include it in the communications system as both a technology demonstration and an inexpensive solution that would provide students with useful skills for the job market. Similarly, the interest in using commercial computing products for space systems has also been growing; so, as part of the LionSat flight computer design, a large amount of commercial SDRAM has been included. The reliability of both the communications system and computer memory is critical to flight operation for LionSat. Because of limited funding and volume for the spacecraft, redundant systems could not be included and hardened systems could not be purchased. Such limitations lead the satellite design

team to pursue in-house development of a reliable communications and memory protection system. The following is a description of the requirements for these systems.

### *Communications System Requirements*

#### 1. Basic Arrangement

Using IP can allow engineers to set up spacecraft or even individual instruments on a payload as nodes on a network that can be accessed with web-based tools, much like the average PC user can access multiple machines over the internet. The popularity of IP in the commercial sector also means that there are plenty of software resources, cheap hardware, and skilled engineers available. There are also several educational advantages that the LionSat project hoped to take advantage of through the inclusion of IP technology in the space-ground link.

During the design process for LionSat, the engineering team determined that the spacecraft flight computer would be capable of running the Linux operating system and that it should interface with ground station equipment as though it were a node on a terrestrial network.<sup>1</sup> More specifically, the engineering team determined that the primary means for accessing the satellite for any purpose should involve FTP, Telnet, and/or a commercial web browser. The team further determined that the LionSat flight computer should be the mediator between a user and the spacecraft instruments and that it should be able to accomplish data storage and forwarding in an automatic mode using common file formats. The ultimate networking arrangement developed for the space-ground link is a layer arrangement with FTP, Telnet, and MDP applications at the top; TCP and UDP for the transport layer; and IP for the network layer. Within the spacecraft, only the computer will have an IP address and the computer will process all information and commands related to onboard systems.

#### 2. Through-put, Encoding, and Synchronization

Analysis of the data transfer requirements for LionSat including overhead show a need to have a downlink capable of 200 kbps and an uplink of 9.6 kbps. The uplink data is intended to be small text files for mission profiles and commands issued by a user; therefore, guaranteed high throughput is not critical. Since the downlink is at a much higher rate, and the information is more precious, greater care must be taken to ensure a high throughput. In addition, there is the desire to more fully realize the “single node on a network” concept through the use of minimal

conversion efforts to transfer packets to and from a standard terrestrial network. Thus, both the up and downlink packets should be in a standard form that is small (for the purpose of minimizing data loss from packet loss), can be easily routed through commercial networking hardware, and can be transmitted over the wireless space-ground link.

The nature of the data to be sent to and received from LionSat is such that there can be periods where no data is available to be transferred, though maintenance of a link is required. For example, a user may send a request for a file, which then takes time to process before it actually enters the downlink pipe. During the periods with no data, maintaining a constant bit stream on the downlink, in particular, is critical to ensuring that ground station receiving equipment does not lose bit synchronization. To prevent loss of lock, and thus loss of packets due to the reacquisition time, the downlink must be encoded such that a clock is buried in the data and the downlink bit stream is effectively continuous. This may be ignored for the uplink for reasons that will be discussed below (Section 3.2.1).

### 3. System Resource Requirements

The minimum system resource requirements for the communication system are simple. Both up- and downlink hardware must interface with the flight computer using either standard serial bus protocols already available on the computer (RS-232, SPI, etc.) or through a simple custom interface involving no more than five general purpose I/O lines available on the flight computer processor. The communications software must be capable of accomplishing the desired IP packetization and the deconstruction of such packets, without loss of data due to delay, on the 200-MHz flight computer running the Linux OS design for LionSat.<sup>5</sup>

#### ***Memory Protection System Requirements***

The effects of radiation on microelectronic circuits can be significant even in a relatively benign orbit. Over long periods, the total accumulated dose can lead to device breakdown causing a steady decline in performance and eventually total failure. In the short term, several types of upsets can also occur resulting in everything from temporary minor glitches to devastating events that result in permanent part or complete system failure. Of particular interest for the scope of this paper is the corruption of information within high speed commercial-quality SDRAM. The deposition of charge by space radiation within a memory cell can cause a bit flip resulting in a

corruption of instructions or data stored in memory. In some cases, a memory device can suffer a latchup—the locking of a memory cell to a fixed state—that requires the device’s power supply to be cycled.<sup>6</sup> During the design process for the LionSat flight computer, a thorough search of both the GSFC\* and JPL† radiation-tested parts databases was conducted to find commercial SDRAM parts with known radiation responses. As a result of the search, it was determined that a system would have to be incorporated into the flight computer to help mitigate the effects of radiation on the SDRAM by detecting and correcting bit errors. In addition, it was determined that the memory protection system should also include the capability to deal with latchups and over-current events that could be solved by cycling power to a memory chip. The ultimate goal of the system is to extend the time between necessary computer resets to as long as possible while protecting science data in memory at the same time.

## **THE SYSTEM DESIGNS**

### ***Communications System***

The LionSat project is a combination of several science and communications missions being performed on one platform. Though the primary mission of the satellite revolves around the two main science goals, the satellite-ground station system also includes experiments in IP technology and software defined radio (SDR) research. As a result, the communications system is somewhat asymmetrical, as shown in the following sections.

#### 1. The Software

The satellite will use IP-based communications with the ground station using the Point-to-Point Protocol (PPP). PPP has become the industry standard for tunneling IP traffic over a direct, two way (duplex) serial link. It also provides a number of other useful features like the assignment of IP addresses, link-quality testing, authentication, and HDLC-based packets. The HDLC (High-level Data Link Control) packets used by PPP offer minimal overhead (5 bytes or less) with two or four bytes of error detection. Existing implementations of PPP are designed to use a single, duplex serial device. This is a problem because the satellite’s RF communications provide

---

\* GSFC Radiation Effects & Analysis Home Page: <http://radhome.gsfc.nasa.gov/top.htm>

† JPL Office 514, Electronic Parts Engineering, [http://radnet.jpl.nasa.gov/cgi-win/1/ViewData\\_CGI\\_Project?|select](http://radnet.jpl.nasa.gov/cgi-win/1/ViewData_CGI_Project?|select)

two separate one-way paths: transmit and receive. Therefore, a special driver is required to provide PPP with a standard serial interface to the RF system. This driver has the benefit of being transparent to PPP, so it is capable of providing additional functionality, such as minimizing packet overhead by removing unneeded information. Once the PPP connection is established, the satellite will be able to use standard IP communications and programs like FTP servers and remote administration applications.

## 2. The Uplink

Early designs for the LionSat flight computer included a commercial FPGA, which was used to implement a transceiver system for the up- and downlinks. One of the primary goals was to create a simple software-enabled radio onboard the satellite that could process uplink data from the receiver into a usable digital form and to convert raw downlink data into an encoded form for transmission. As the design process progressed, a choice was made to forgo the use of the FPGA for the uplink receiver in favor of a pair of Chipcon CC2400DBK transceiver modules. These modules were chosen because they contained both the RF and digital hardware necessary to create the desired wireless link using a simple RS-232 interface. The units can provide sufficient throughput, accept and produce data in byte form, and can be adjusted to the desired frequency band with some simple RF front-end hardware. To complete the uplink, the LionSat software engineer created a modified network driver to allow the flight computer and a ground station desktop computer, both running Linux, to perform IP communications through an RS-232 port. The resulting uplink configuration is such that packets on the ground station computer are chopped into HDLC frames and fed via an RS-232 port to a transceiver board, where they are converted to an RF signal and transmitted to the satellite. Onboard the satellite, a Chipcon transceiver accepts the RF signal, converts it to digital form, and feeds the data one byte at a time to the flight computer over an RS-232 port. The supporting software on the flight computer performs the necessary reassembly of the HDLC frames and conversion to useable IP packets. HDLC frames are used in the uplink only for consistency in the overall data arrangement. Their real usefulness is demonstrated in the downlink description.

## 3. The Downlink

The downlink is an adaptation of the standard seven-layer networking scheme to include the wireless space-ground link. Starting at the top, a common

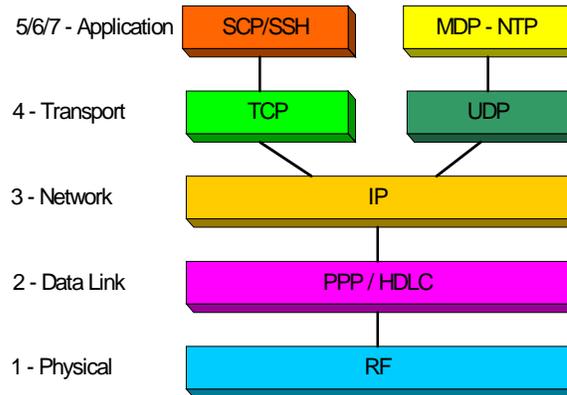
software package such as an FTP server is kept running on the satellite flight computer in a Linux environment, and standard networking protocols are used all the way down through layer three (network layer). Layer two (data link layer) involves a conversion of IP packets to HDLC frames, again in software. HDLC frames were chosen based on the advice of the NASA GSFC OMNI (Operating Missions as Nodes on the Internet) team, because there is plenty of inexpensive commercial networking hardware that can accept HDLC frames for conversion to a regular terrestrial LAN. The small size of the frames also provides two additional advantages. It helps to reduce the possibility of data loss from signal damage and HDLC frames can easily be substituted with empty frames to maintain a ground station bit-sync during periods of lack of IP packets for transmission. The ease of substitution is critical to simplifying the ground station recovery hardware. As was previously mentioned, the LionSat communications system includes an SDR experiment that is being used as the ground-station downlink receiver. If only the transmitter carrier were available to the receiver between HDLC frames, then the ground station would have to try and reestablish bit synchronization when valid data begins again. This could result in a loss of the initial packets. Maintaining bit-synchronization by transmission of filler frames during “gaps” increases the reliability of the downlink and allows the SDR system to be implemented with a simple phase locked loop (PLL).

Once the HDLC frames are formed, they are fed through a 5-wire bus to logic in the FPGA included in the flight computer, where they are mixed with a clock through Bi-phase-L encoding and then fed to an FSK transmitter at the last layer (physical layer). Figure 2 provides a block diagram of the network layering.

The FPGA used in the flight computer is a radiation-hardened Eclipse FPGA donated by Aeroflex, Inc. It is radiation hardened up to a total ionizing dose (TID) of 300 krad(Si) and has a latch-up immunity up to 100 MeV·cm<sup>2</sup>/Mg.\* Using this FPGA is particularly advantageous to the LionSat team because it provides a platform for implementing the communications encoder and the memory ECC, described later, that is protected against radiation effects.

---

\* Rad-Hard ECLIPSE family FPGA listing for Aug 2004, <http://ams.aeroflex.com/ProductFiles/DataSheets/FPGA/Ra dHardEclipseFPGA.pdf>



**Figure 2: Network layering block diagram for LionSat space-ground communications.**

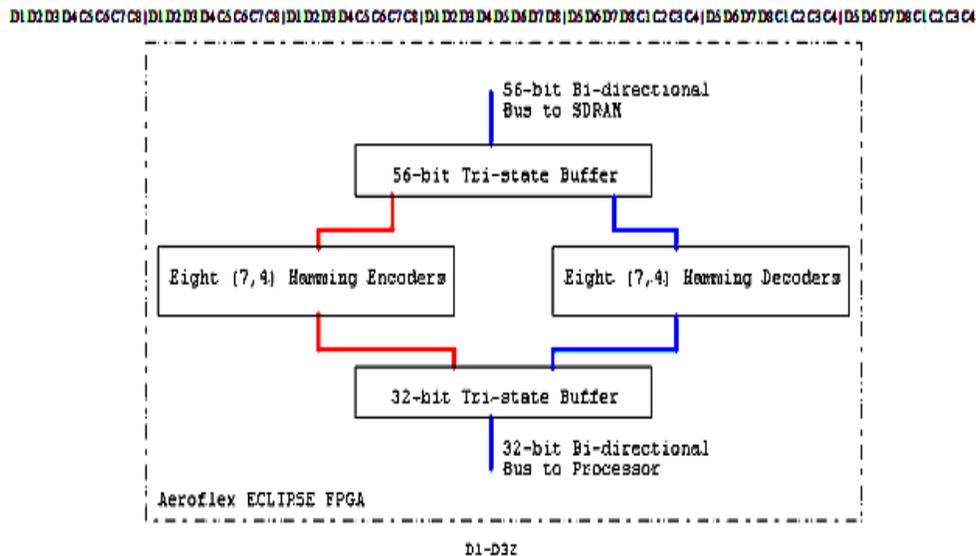
The logic in the FPGA for the downlink path is designed to serve two purposes. The first is the encoding of HDLC frames received from the processor into a Bi-phase-L data stream that can be fed directly over a one-wire bus to the transmitter. The second is to ensure that when valid frames are not available from the computer, filler frames are regularly stuffed into the encoder to maintain a constant bit stream.

Figure 3 is a block diagram of the communications encoder for the LionSat downlink. There are two main sections. The input portion consists of a serial-to-parallel shift register that receives HDLC frames from the computer. The output portion comprises a parallel-to-serial shift register driven by a 200-kHz clock and linked to a counter. In order to establish a downlink the computer first drives the transmit-enable pin, which engages the clock and output portion of the logic. As the clock runs, the contents

of the parallel-to-serial register are shifted into an XOR gate, where it is combined with the clock signal creating a Bi-phase-L formatted data stream. A counter connected to the clock and latch line of the shift register ensures that every time the shift register makes a complete cycle through its contents, the data applied to its input pins are latched and the RDY FPGA signal is set. The first four bytes of the shift register contents will be fixed to a specific pattern that will act as an identifier to help the ground station maintain a lock.

On the ground, hardware will recover the HDLC frames and construct Ethernet packets from them. This conversion can be accomplished with a commercial router. The Ethernet packets will then be fed onto a LAN where they will be received by the ground station computer. The flight computer will be the only owner of an IP address on the spacecraft.<sup>3</sup>





**Figure 4: Block diagram of the error correcting arrangement for protecting commercial SDRAM against eight independent bit errors. At the top, D1 to D8 represent a bit from each of the four 4-bit chunks and C1 to C8 represent a bit from each of the eight sets of check bits.**

## 2. An SDRAM ECC Design with Back-up Memory

The ability to correct eight independent bit errors is particularly useful because the bits in the 56-bit word output to the SDRAM can be arranged such that if an entire 8-bit SDRAM chip is lost, then the Hamming decoder can correct for it and still recover the 32-bit data word. This, of course, is true only if additional bit errors do not occur in other RAM chips. The proper arrangement of the bits is shown at the top of Figure 4. One bit from each 4-bit data chunk and one bit from each 3-bit check set is routed to each RAM chip. Noting the difference in the number of check and data bits, the reader will realize that one of the RAM chips will have only data bits routed to it and only 7 chips will be needed per bank instead of the eight needed for the simple ECC implementation. With this arrangement a loss of an entire chip still only causes one bit error per grouping.

To properly accomplish the scrambling, a write to SDRAM requires that the logic inside the FPGA be arranged such that it first reads the 32-bit word from the address to be written to, the bits to be preserved are masked, if any, and the new word is then written into RAM. This complicated procedure is necessary because the SDRAM controller in the SA-1110 processor, as with many other processors using SDRAM, implements a masking signal automatically when writing less than 32 bits to memory. The

masking signals are labeled as lines DQM0–3 on the processor and are normally distributed to 8-bit chunks of SDRAM. These signals do not appear to have an override to prevent their use; thus, if an operation occurs that requires only 8-bits in a memory location to be changed, the scrambled RAM arrangement would cause a problem since the 8 bits are distributed throughout all the chips.

The occasion of loss of an SDRAM chip may not be due necessarily to the chip failing completely. Radiation exposure can cause non-hardened parts to increase their current draw over time or to latch up. These situations can often be corrected by simply resetting the power to the affected part. As a precaution against excessive accumulation of increased currents and latchup, the SDRAM chips may be equipped with a small “smart” circuit breaker. When applied in combination with the data bit scrambling, a backup memory chip can be included on the memory bus in parallel with each primary SDRAM chip. Two circuit breakers can be arranged in pairs for every SDRAM chip and its corresponding back-up so that if a circuit breaker trips on a primary part, it then turns on the power to the back-up part. The newly-engaged 8-bit part, though empty, will be reloaded with data from every write cycle and each read cycle that provides invalid data will be overcome by the 8-bit ECC.<sup>7</sup>

## CONCLUSION

At the time of this writing both the communications system design and the simple memory error correcting code design have been successfully created in VHDL and simulated. The combined total resource requirements for both systems uses less than 45% of the logic cells available in an Eclipse FPGA. Future work will involve the implementation and testing of both circuits in separate units followed by testing of a combined implementation in one FPGA.

## ACKNOWLEDGMENTS

B.S. Surrusco thanks Aeroflex Colorado Springs, Inc. for their gracious support of LionSat computer hardware requirements and the NASA GSFC OMNI team for their guidance concerning the implementation of IP over the LionSat space-ground link.

LionSat Project Sponsors include: Aeroflex Colorado Springs, Inc., Air Force Office of Scientific Research, Air Force Research Laboratory Space Vehicles Directorate, Boeing, C&R Technologies, Lockheed-Martin, National Aeronautics and Space Administration (GSFC), Penn State Aerospace Engineering Department, Penn State College of Engineering, Penn State Electrical Engineering Department, The American Institute of Aeronautics and Astronautics, and The Pennsylvania Space Grant Consortium.

## REFERENCES

1. Mistoco, V. F., R. D. Siegel, B. S. Surrusco, E. Medoza, and S. G. Bilén, "Design of the Local Ionospheric Measurements Satellite," 17th Annual, AIAA/Utah State University Conference on Small Satellites, Logan, UT, August 2003.
2. Mistoco, V. F., S. G. Bilén, and M. M. Micci, "Development and Chamber Testing of a Miniature Radio-Frequency Ion Thruster for Microspacecraft," 40th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit Fort Lauderdale, FL, July 2004.
3. Rash, J., and Hennessy, J., "Implementation Guide for the Use of the Internet Protocol Suite in Space Mission Communications," Draft Release 1.0, Information Systems Division, NASA Goddard Space Flight Center, Greenbelt, MD, Sept. 2003.
4. Surrusco, B. S., Siegel, R. D., Sorber, P. M., Morr, D. V., O'Connor, N. F., Croskey, C. L., Bilén, S. G., and Soloff, J. A., "Mission planning for employing internet protocol (IP) communications on the Local

Ionospheric Measurements Satellite (LionSat)," NASA Third Space Internet Workshop, Cleveland, OH, June 2003.

5. O'Connor, N. F., B. S. Surrusco, C. L. Croskey, and S. G. Bilén, "Software-Defined Radio Ground Station for Internet-Protocol Communications to a Low Earth Orbit Nanosatellite," NASA Fourth Space Internet Workshop, Hanover, MD, June 2004.

6. Schneider, W., G. Gardner, E. Marian, et al, IEEE Standard for Environmental Specifications for Spaceborne Computer Modules, IEEE Std 1156.4-1997, pp. 1, 14-16, 22-27, 29-30 July 1997.

7. Surrusco, Brendan S., A Low Cost, Powerful Flight Computer Design Including Linux and Internet Protocol Technology for the Low Earth Orbiting Local Ionospheric Measurements Satellite, M.S. Thesis, Electrical Engineering Department, The Pennsylvania State University, University Park, PA, May 2005.