

An architecture for a space-based reconfigurable protocol chip

Clayton Okino^{*}, Clement Lee^{*}, Andrew Gray^{*}, Payman Arabshahi^{*}, Wai Fong⁺, Jason Soloff⁺

Jet Propulsion Laboratory^{*}
California Institute of Technology
4800 Oak Grove Drive M/S 238-343
Pasadena, CA 91109

[\[Clayton.Okino, Clement.Lee, Andrew.Gray, Payman.Arabshahi\]@jpl.nasa.gov](mailto:{Clayton.Okino, Clement.Lee, Andrew.Gray, Payman.Arabshahi}@jpl.nasa.gov)

Goddard Space Flight Center/NASA⁺
Code 567
Greenbelt, MD 20771

{Wai.Fong, Jason.Soloff}@gsfc.nasa.gov

Abstract - In this work, we present a framework for a space-based reconfigurable protocol chip. We identify key input stimuli that would promote reconfiguration. Utilizing the identified stimuli, we then present a basic architecture that enables configuration. Significance is placed on the protocol aspect and in particular, detection and reconfiguration of a link layer protocol. We present a sensing mechanism that detects when link layer reconfiguration is required and finally present results on the likelihood of false reconfiguration for our link layer protocol sensing technique. In addition to the sensing aspect, fault tolerance is addressed.

I. INTRODUCTION

An architecture for implementing a software reconfigurable network processor for satellite communication applications is presented. The reconfigurable protocol chip enables rapid *autonomous reconfiguration* of space communications network functions. This reconfiguration provides long-life *space communications infrastructure*, enables *dynamic operation within space networks with heterogeneous nodes*, and *compatibility between heterogeneous space networks* (i.e. distributed spacecraft missions using different protocols) as depicted in Figure 1. This work builds upon numerous advances in commercial industry as well as NASA and military software radio developments to develop reconfigurable space network processing and processors. The development of such radios and the network protocol chip presented here require defining the correct combination of processing methods and developing appropriate dynamic reconfiguration techniques as a function of system goals and operating parameters. Dynamic reconfiguration techniques developed herein include autonomous network/protocol identification and autonomous network node reconfiguration. Both the Earth Science Enterprise Strategic Plan and Research Strategy for 2000-2010 identify satellite constellations and specifically distributed spacecraft and particularly formation flying technologies as an important technology thrust and investment areas, applicable to a range of missions. Such missions will have wireless network protocols derived or extended from commercial efforts in this area. Specifically, commercial protocols might be used or might be modified

for use in many future distributed spacecraft missions. It is a tremendous challenge to find one “universal” protocol to meet the requirements of all of these future missions; in fact the challenge is so great that it is the belief of these authors/investigators that there is not such a protocol in existence (particularly one that is commercially available) and that even if such a protocol existed it’s complexity would have serious drawbacks from realization/implementation and adoption perspectives. This being the case, missions in the next 5-10 years are extremely likely to be operating with multiple protocols and substantial protocol variations depending on the requirements of the distributed spacecraft mission. The reconfigurable protocol chip seeks to reduce the cost and risk of future mission by enabling internetworking compatibility and on-mission reconfiguration; therefore networks with more general resource sharing and backward /forward compatibility can be maintained.

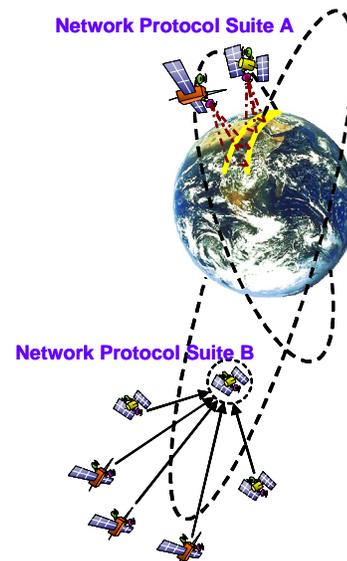


Figure 1 Heterogeneous networks in space

II. FRAMEWORK AND ARCHITECTURE

In this section, we describe the basic reconfiguration architecture for space-based applications. We identify the

key input stimuli, the mechanism that perform detection of the stimulus and some processing either coupled or decoupled that executes intelligent decision on the input.

As depicted in Figure 2, the reconfiguration architecture presented contains three key components required to identify and perform reconfiguration in space: (1) External stimulus detected will either result in a requirement to perform a chip reconfiguration or a desire to reconfigure a chip; (2) Sensors are required to detect and possibly perform specific in-situ processing on the detected stimulus; (3) Intelligent processor performs the decision either independently or in a coupled manner if multiple stimulus are correlated such that desired outcomes of the process have differing reconfiguration mappings.

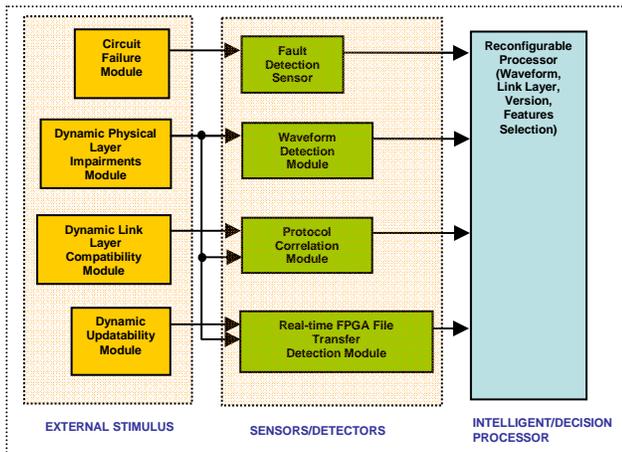


Figure 2 Space-based Reconfigurable Chip Architecture

A. External Stimuli

In space-based operations, various interactions are desirable or necessary. We identify a set of currently realizable or desirable sensing interactions such as radiation, physical layer communication, link layer variability (inter-heterogeneity and intra-heterogeneity), updatability (to improve overall performance or correct errors in original design).

1) Radiation

A key source of failure of a module in space resulting in a system fault in space environment [2]: Galactic Cosmic Rays (GSM), Solar Radiation (e.g. Solar Wind/Protons, Coronal Mass Ejections), Planetary Magnetic Fields (e.g. Van Allen Belts, Jovian belts). Some key types of radiation effects [2][3] are Total Ionization Dose (TID): cumulative ionization causing increase in leakage current and threshold shifts; Single Event Effects (SEE): single particles, Linear Energy Transfer, Single Event Latch up (SEL), Single Event Upset (SEU), Single Event Multiple Upset (SEMU), Single Event Gate Rupture, Single Event Micro-dose

2) Physical Layer Communication Impairments

In space, key impairments and the effect it has on performance at the physical layer (assume RF links) are due

to variations in the channel. The effect of these impairments can be mitigated utilizing various waveforms, error correction techniques, (as well as link layer reliability techniques and other higher layer interactions). Ideally one could map EIRP (perform link budgets, map to a potential set of waveforms or allowable waveforms, then perform appropriate detection in the Waveform Detection Module). Beyond a brief description of the Waveform Module description is techniques is beyond the scope of this work and is out of scope of this paper although is popular among the Software Radio community.

For our approach and for the remainder of this paper we assume a single waveform specifically, a BPSK waveform.

3) Link Layer Communication Interaction

In terms of OSI layer 2, we recognize that space-based variability in terms of reconfiguration amount to the possibility of a number of link layer protocols and the ability to interact among various heterogeneous networks. If we assume a synchronization capability of some form either octet synchronous possibly due to framing performed at the forward error correction framing level, or some other mechanism such as described in the Goddard Space Flight Center (GSFC) Parallel Integrated Frame Synchronizer (PIFS) Chip, we can then perform additional framing detection as can be found in many standards. As a baseline capability, we assume a form of HDLC (RFC1662)[4] & 802.3 link layer framing.

4) Reconfiguration Variability

Version upgrades, added features, reliability of valid transfer are all desirable and in some cases required mechanism in a reconfigurable platform. Analogous to this philosophy is the ability to perform upgrades and add software while a spacecraft is in transit to a remote location. Some preliminary work has been performed for a spacecraft avionics architecture to provide reconfiguration in-situ [5]. In particular, [5] describes mission operation procedures and uplink/downlink process to reconfigure the spacecraft in-orbit where commands are defined to execute the in-flight hardware reconfiguration where spacecraft safety is of significant concern.

B. Sensors/Detectors

We now describe the sensors capabilities and partial processing required for our proposed reconfiguration platform.

1) Fault Detection Sensor

The fault detection sensor must detect and distinguish between transient and permanent faults. The trade offs for different methods of fault detection include circuit down time, circuit complexity, and detection update rate. The chosen, low complexity fault tolerance scheme allows for a basic level of reliability. Other complex fault tolerance schemes can be implemented on top. In order to constrain the

complexity of our paper, we omit the complexity of other fault tolerance schemes¹.

Cyclic Redundancy Check (CRC) codes provide a simple and effective tagged data scheme to monitor data corruption in many applications. CRC codes were selected as the fault detection sensor scheme due to their ease of implementation. However, system downtime and detection update rates may be an issue depending on the application, i.e. network latency requirements. In our scheme, CRC codes will be inserted into the data processing periodically. A single CRC failure will trigger a transient fault detect. Multiple consecutive CRC failures will trigger a permanent fault detect.

2) Waveform Detection Module

Many variants on dynamic waveform detection and reconfiguration can be employed. In particular the ability to detect a particular waveform in-situ and then reconfigure in space is a novel concept even though the basic concept can employ well-known techniques as are used in standard dialup modems. For our case, we shall assume that the waveform is BPSK and leave the remainder of waveform detection for future work out of the scope of this paper

3) Protocol Correlation Module

The protocol correlation module is a layer 1 sensor that is expected to detect between a set of possible protocols. The concept of heterogeneous networks in space will be driven by a number of variables outside of the scope of this work. However, we can consider target protocols that have high probability of use in future space-based networks. Among these are HDLC variants (e.g. RFC1662), 802.3, and GFP. For this paper, we consider two of the protocols, 802.3 and RFC1662 and focus on some of the detection variants for RFC1662.

4) Real-time FPGA File transfer Detection Module

Beyond the standard mechanism employed in COTS Network Interface Cards (NICs), the concept of updating the file transfer process at the link layer is a novel concept that has not been fully leveraged. In particular, for FPGA based transmissions as is well-defined for Xilinx operation, one can employ a real-time reliable mechanism that will allow for fast Xilinx reconfiguration file transfers. With the advent of the higher density higher speed RAMs, we can employ faster Layer 1 processing for FPGA as well as other file transfer mechanism as depicted in Figure 3. This concept was presented in [6] for a general reliable file transfer and is basically leveraging the added capabilities that are being designed into NICs.

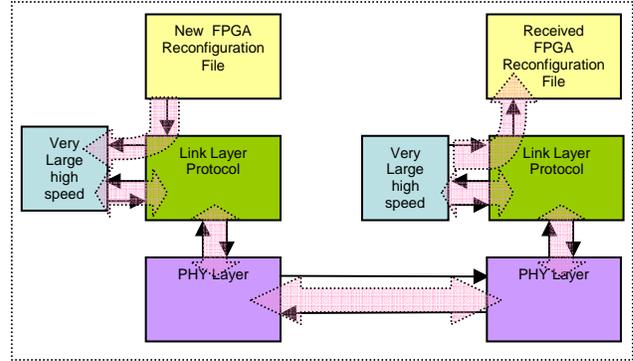


Figure 3 Link Layer Reconfigurable file transfer platform

As part of this architecture described above in Figure 3, we can employ robust protocols such as hybrid ARQ [7] techniques in addition to file checking at the link layer rather than traditional end-to-end verification.

III. FAULT TOLERANCE & LINK LAYER PROTOCOLS AND SENSING

In this section, we focus on two of the four sensing areas for reconfiguration. Specifically, we examine processing algorithms and techniques for fault tolerant mapping and link layer recognition and processing schemes.

A. Fault Tolerant Mapping

The reconfigurable processor uses a simple two tiered fault diagnosis and recovery architecture, as shown in Figure 4.

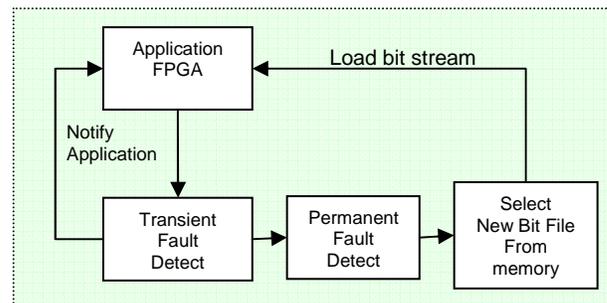


Figure 4 Two tiered fault detection and recovery

Transient fault detects will be accumulated to detect permanent faults. Other than notifying the application, nothing else will be done to correct transient faults. We can rely on higher layers to deal with the effects of transient faults. Permanent faults will be dealt with by reprogramming the FPGA with an alternate pre-compiled spatial variant of the same application.

The motivation for this fault tolerance mapping is to increase the dependability of the link layer operation within the FPGA. The overall architecture is based on a priori knowledge of the failure within the FPGA.

¹ In general, we anticipate utilizing schemes such as Triple Modular Redundancy (TMR), a form of circuit replication and voting for space-based fault tolerance.

If the spatial representation of an FPGA is defined in Euclidean coordinates (x,y) , then let $p(x,y)$ be the probability of a point in an FPGA failing. Given this distribution, we could simply constrain our FPGA design to minimize use of the points with the highest values of $p(x,y)$. The probability of failure for the i th configuration file occupying some subset, r_i , of the entire FPGA is

$$P_i = \sum_{(x,y) \in r_i} p(x,y)$$

To simplify the problem, our pre-compiled designs will be constrained to equal spatial distribution orientations. The probability of failure for any configuration will be the same. Figure 5 shows an example circuit with 75% utilization, constraining the circuit to 3 of the 4 quadrants. The probability of failure is 1/4.

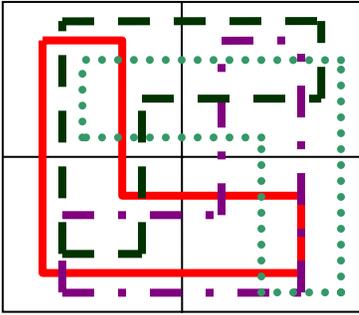


Figure 5 Redundant spatial variation of FPGA utilization

The spatial variants for a protocol detection circuit for 802.3 and HDLC framing structures is shown in Figure 6. Due to the relatively small size of the design, the Xilinx Spartan 2 XC2S30 FPGA was selected, where utilization is ~ 50% [1].

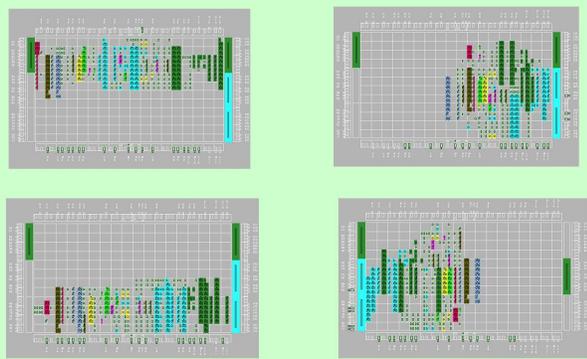


Figure 6 FPGA Floor Plans for 4 phases of constraints with 50% utilization

B. Link Layer Recognition and Processing Schemes

We assume that the physical layer is octet synchronous for both the 802.3 frame structure and the RFC1662 HDLC frame structure. Specifically, the 802.3 preamble is omitted and we focus on the 802.3 start frame delimiter and the HDLC opening flag. As in any link layer protocol some of the primary functional attributes are frame

synchronization, addressing, multi-protocol selection, data transparency, and reliability. To simplify the analysis, we focus on the RFC1662. Furthermore, we assume that the address field is set at 8 bits, the control field is fixed, the Frame Check Sequence is fixed at 16-bits and we are not utilizing ARQ.

For frame synchronization, it is straightforward to perform a cross correlation between the two start field bit sequences. Recognize that 0x7E and 0xAB differ in exactly 5 bit locations as depicted in Figure 5.

Consider a generic threshold circuit that is needed to validate the start flag for a single link layer protocol. In the case of RFC1662 (or 802.3), tolerating a number of bit errors (bit flips) in the start flag would be desired. Recognize that a sensing decision circuit in the form of a threshold decision circuit used to determine if the protocol is 802.3 versus HDLC will make an incorrect decision if at least 3 of the differing bits are in error (i.e. it will mistaken one protocol for the other).

Suppose p is the probability of a bit error. Then among the 5 differing bits, if any 3 or more bits are in error, then the sensing decision circuit will result in a protocol decision error in the form of a binomial distribution as described in the equation below.

$$\begin{aligned} \Pr(\text{False protocol detection}) &= \sum_{i=3}^5 \binom{5}{i} p^i (1-p)^{5-i} \\ &= p^5 + 5p^4(1-p) + \binom{5}{3} p^3(1-p)^2 \\ &= p^5 + 5p^4 - 5p^5 + 10p^3 - 10p^5 \\ &= 10p^3 + 5p^4 - 14p^5 \end{aligned}$$

As depicted in Figure 6, we examine a plot for likelihood of false protocol sensing as a function of Signal to Noise (SNR) for uncoded Binary Phase Shift Keying (BPSK) modulation conditioned on reconfiguration between the two defined protocols using the simple threshold decision circuit.

In Figure 7, we observe that in general, the likelihood of a false sensing and error protocol configuration is low and decreases fast with respect to the bit error rate (BER) for BPSK. However, if the circuit is consistently monitoring on a per packet basis, and a burst of bit errors occur, then invalid reconfiguration could occur on a per packet basis. To reduce the likelihood of “protocol configuration flapping”, we introduce a Markovian state based concept where we condition re-configuration on prior states.

Ideally, we would like the conditional state probability distribution of the sensing error. As an approximation, it would be advantageous to use the conditional average bit error rates.

$$\Pr(\text{error at time } t = T) = \sum_{i=3}^5 \binom{5}{i} p_T^i (1-p_T)^{5-i},$$

where $p_T = p(t = T / t = 0, 1, 2, \dots, T-1)$, the average probability given the probability of the previous bit time slots. In general, one could assume that since all bits are

independent, this is fixed to p , the probability of a bit error. However, if in a space-based (wireless) scenario the channel correlates bit errors (analogous to burst errors), then the independence assumption no longer holds and a conditional distribution is desired for state dependent autonomous protocol reconfiguration. We introduce an example of such an algorithm in [1].

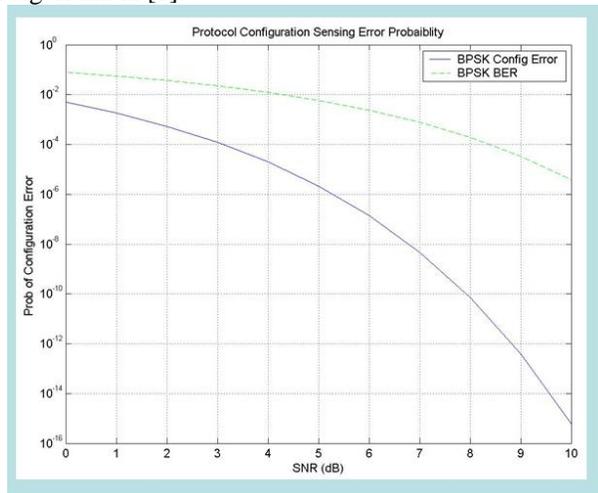


Figure 7 Protocol Sensing Error Probability

We now extend the concept of error detection with higher resolution. Specifically, we consider identifying the data transparency variations within RFC1662. In particular, we detect the difference between the bit-stuff operation (RFC1662 Section 4) and the byte stuffing operation (RFC1662 Section 5). First, we briefly describe these two stuffing mechanisms and then describe a procedure for resolving the stuffing approach being used.

From RFC1662, for the byte-stuffing procedure, the bit sequence is examined on an octet by octet basis. Since the flag sequence is 0x7E and we assume that the likelihood is uniform among all possible octet sequences, we have the well-known result for this sequence occurring with probability $1/256$. Specifically, in RFC1662 the 0x7E sequence maps to 0x7D followed by 0x5E. Another possible character re-mapping is the control escape sequence 0x7D re-mapped to 0x7D followed by 0x5D.

From RFC1662, for the bit-stuffing procedure, the bit sequence is examined on a bit by bit basis. Since the flag sequence is 0x7E (containing five one's in a row), then a "0" bit is inserted after all five contiguous "1" bits. We have the well-known results of the likelihood of these sequences occur with probability $1/32$.

In addition to utilizing the traditional CRC codes to validate that frames are correct, we can also validate using the special sequences described for the byte stuffing procedure. We assume that the only re-mapping for the byte stuffing procedure are the flag sequence and the control escape sequence. If we assume that the control escape sequence is almost never used, then we are evaluating if the bits sequence 0x7D5E exist versus the bit sequences that

equates to inserting an additional "0" using bit stuffing equating to the 15-bit sequence "01111101011110". The likelihood that this is originally a bit stuffing process would be the likelihood that this exact 15-bit sequence occurred resulting in a probability of $1/2^{15} = 3e-5$. By executing this checking process and then weighting this scenario as a bit stuffed process with the $1/2^{15}$ likelihood followed by the proper CRC based on detecting the end-of-frame correctly then we can select the type of stuffing. Further examination into this the benefits of this procedure as oppose to simultaneously implementation of both stuffing procedures is under investigation. Note that weighting likelihood detections schemes of this form allow for a level of scalability but also present some finite likelihood of false detection.

IV. COMMENTS

We presented a promising architecture—that includes stimuli sensing capability and an intelligent processor—for a space-based reconfigurable protocol chip. We examined a simple strategy for detecting and combating faulty circuitry. Finally, we presented some standard link layer framing protocols and identified a detection mechanism for the data transparency variants in RFC1662. In addition to refinement of the link layer protocol set, there is significant interest in refinement of the reliable link layer file transfer architecture and corresponding protocol.

REFERENCES

- [1] C. Okino, C. Lee, A. Gray, P. Arabshahi, "An Autonomous Evolvable Architecture in a Reconfigurable Protocol Chip for Satellite Networks", 2003 MAPLD International conference, September 9-11, 2003, Washington, D.C.
- [2] John Scarpullla and Allyson Yarbrough, "What Could go Wrong? The Effects of Ionizing Radiation on space Electronics", <http://www.aero.org/publications/crosslink/summer2003/03.html>
- [3] Raphael Some, "Radiation Models and Hardware Design", presentation in 2002.
- [4] RFC1662 – PPP in HDLC-like Framing, July 1994.
- [5] Saio Chau, Adans Ko, Kar-Ming Cheung, "Mission operation for reconfigurable spacecraft", SpaceOps 2004 conference.
- [6] L. Clare, J. Gao, E. Jennings, C. Okino, "Reliable Link Layer File Transfer" DRAFT technical report May 2004.
- [7] Shu Lin, Philip S. Yu, "A Hybrid ARQ Scheme with Parity Retransmission for Error Control of Satellite Channels", IEEE Transactions on Communications, no. 7, July 1982 pp. 1701-1719.