Reliability and Performance Analysis of a Fault Tolerant Data Handling Protocol for Aerospace Applications

R. Omidi[1], A. Eidi[2], L. Mohammady[2] and S.Y. Torabi[2]
[1]*Zanjan University, Iran*
[2]*Satellite Communication Group, Iran Telecommunication Research Center (ITRC)*
omidi.g.r@gmail.com

***Abstract*—Data communication inside the satellite is one of the most important factors in satellite design. For this purpose, a variety of protocols have been developed in recent years. Controller Area Network (CAN) is one of the well-developed protocols to be used in the On-Board Data Handling (OBDH) systems for communication and geosynchronous satellites. Nonetheless, for aerospace applications which demand radiation hardened integrated circuits, a full featured stand-alone Rad-Hard CAN controller is unavailable. HDL (Hardware Description Language) based IP (Intellectual Property) Cores which are widely developed to be implemented on Rad-Hard FPGAs are more attractive. This paper proposes a novel fault tolerant CAN controller based on FPGAs to provide on-board data handling requirements of the communication satellites. We outline some practical topologies and discuss their complexities and reliability. Despite the fact that the most famous methods like TMR (Triple Modular Redundancy), are very common among designers, the reliability analyses show that these methods are unable to tolerate single upsets in routing matrixes. This paper proposes a robust data bus controller based on dual duplex redundancy on FPGAs. The fault injection experiments reveal that the proposed approach represents better performance respective to the conventional hardware redundancy. Furthermore, the experiments show that the capability of tolerating SEU effects by the proposed method is increased up to 7.17 times with respect to a regular design. The proposed architecture imposes 16.26% and 5.2% overhead in the required resources and the operating frequency in comparison to the regular TMR method.**

***Index Terms*— Communication satellite; Data Handling; Space Radiation Faults; Reliability; Controller Area Network (CAN).**

I. INTRODUCTION

Many of the wireless services offered today are based on the research efforts initiated by satellite communications scientists decades ago [1]. Satellite technologies have always played a key role in providing widely covered communications for ordinary as well as emergency situations. In order to have a successful satellite communication, we have to provide appropriate systems and facilities in ground stations, wireless data link, and satellites on orbit as shown in Figure 1. Modern communications satellites use a variety of orbits such as geostationary orbits. Most commercial communications and broadcast satellites operate in geostationary orbits. The space radiation environment in GEO (geosynchronous) orbits is very dynamic, and satellite design for this orbit requires more challenges. Many works have been developed in the literature to partially solve this issue. A review of previous works found several reliable designs, such as fault tolerant filters, multipliers, coders and decoders, which are developed for satellites. In this paper, we aim to propose a fault tolerant data handling protocol that can be used for GEO satellites.

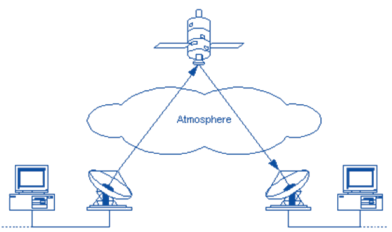

Figure 1: Satellite communication

Data handling and data processing inside the satellite can be implemented centralized or distributed. For a space system in which data handling is based on centralized methods, a central control unit is responsible for performing all the data handling and management tasks. This approach will result in a great amount of wiring, high complexity and weak reliability. In new spacecraft, the architecture is moving from fully centralized towards distributed processing and data handling via communication protocols [2]. There is a wide range of communication protocols that can be used in communication satellites, ranging from the extremely common interfaces such as RS-485 and I2C to the custom protocols implemented by individual vendors and application developers [3]. Currently, serial communication standards used for space applications are mainly based on MIL-STD-1553B and RS-485 which their inherent characteristics such as Master/Slave or Client/Server configuration make them inefficient to handle requirements of new systems (e.g. redundancy support) [3-5]. In the case of MIL-STD-1553, another withdraw is the high power consumption of interfaces that makes it a non-affordable solution for power restricted applications [5-6].

The CAN protocol is an international standard defined in the ISO 11898-1 standard based on the "broadband communication mechanism" which is applied with a message-oriented transmission protocol as shown in Figure 2 [7]. This protocol was initially created by German automotive system supplier Robert Bosch in the mid-1980s specifically for automotive applications to make them more reliable, safe, and fuel efficient while at the same time to decrease wiring-harness weight and complexity. Due to these advantages, the CAN protocol is also a good candidate for other areas such as aerospace, maritime, railway vehicles, industrial automation and medical equipment [3].

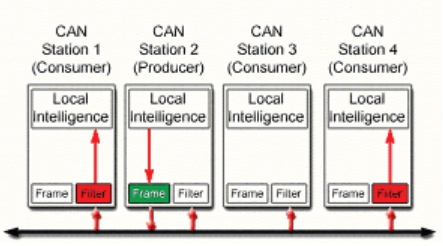

Figure 2: CAN protocol data transmission

The first successful example of CAN usage in ESA missions returns to the SMART-1 satellite in the late '90s [8]. However, mostly because of the widespread adoption of MIL-STD-1553B, it has taken some time for the CAN bus to get real traction in Space community. The research and activities performed by Thales Alenia Space Italia (TAS-I) demonstrated the CAN bus as an effective solution for both scientific and SATCOM missions [9]. Recently developed, the ECSS-E-50-15C standard "CAN bus extension protocol" extends the CAN bus specification to cover the aspects required to satisfy the particular needs of spacecraft data handling systems [10]. The CAN IP cores have been used in some notable ASIC developments (GR712RC, UT699/UT700, AT9713E, COLE) as well as in custom FPGA developments targeting missions such as the Eurostar E3000 for telecoms, Sentinel 1 for Earth observation, and the Exomars rover for science and space exploration, the International Space Station (ISS) etc. However, the ASIC design of a fault tolerant CAN IP core has not been developed yet. The FPGA based HDL core is developed and evaluated. When these HDL IP core is completely developed, their new features will also be included in all future rad-hard ASIC developments.

In this paper, we aim to develop an IP core for CAN controller based on FPGA. The proposed core has been hardened through the fault masking and diagnosing methods. Also, the fault model is related to space radiation in satellite environments.

There is a considerable of literature on CAN controller implementation based on FPGA [11-13]. Some preliminary works were carried out in the early 1990s, while the continuing revolution in FPGA technologies demands newer designs. Also, the harsh space environments force the designers to provide the hardening techniques.

In satellites, when a charged particle in the space environment strikes to an active area of an integrated circuit, it generates a transient current pulse that can cause unwanted effects. This phenomenon is able to produce an inversion in the stored value in the memory cells and registers. A bit flip in the memory cell is called single event effect (SEU), and it is one of the major concerns in SRAM-based FPGAs. An effective strike can also provoke multiple bit upsets (MBU) in the registers and transient effect (SET) in microelectronics. Triple modular redundancy (TMR) is the most frequently used mitigation technique in which the main module is triplicated and through a majority voter, the valid output is selected. In order to use this approach on FPGA, some pitfalls should be considered [14] to achieve an accurate scheme. Also, to verify the fault masking ability of the mitigation method, the fault injection methods are used in the literature [15].

This paper aims to provide an area efficient, fault tolerant structure for CAN controller on SRAM based FPGAs. Firstly, we represent our HDL based CAN controller which is fully tested and validated by hardware. Then we investigate hardening techniques and introduce dual duplex structure to mitigate SEU faults. We analyse the reliability of the hardening methods and show that with an appropriate coverage factor, the dual duplex provides better dependability. Finally, through the fault injection platform, we tested the hardening techniques against single and multi-upset faults. We have used some features like CRC check and bit stuffing capabilities in CAN protocol to achieve better coverage factor. Based on reliability analysis and fault injection results the proposed dual duplex method provides better dependability, although it requires more resources. In the following section we detailed the fault model and mitigation technique.

II. Fault Model and Mitigation Techniques

The basic internal structures of the FPGAs are very similar. Normally, a sort of configurable logic blocks (CLBs) has been integrated with programmable switch matrixes that connect the logic blocks according to the design requirements [14] (0). The user required logics are synthesized in CLB blocks, these blocks are routed through the switch matrixes to provide the user desired function. The logic mapping and routing scheme are completed through a sort of programming bits, which are called bit stream. According to the FPGA technologies, the bit stream is stored in SRAM memory, Flash memory, or directly set a sort of fuse. In case of SRAM based FPGAs, which are more attractive for designers, we encounter a serious challenge for aerospace applications.

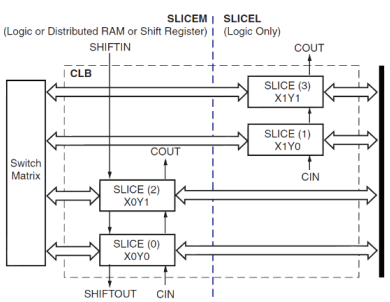

Figure 3: Virtex-4 FPGA architecture [14]

All logics in slices are based on look-up tables, which are the same as SRAM memory. Also the interconnections in switch matrixes are controlled through the SRAM based memory cells. The basic structure of the switch matrix and single event upset effect is illustrated in 0, more details have been discussed in [14].

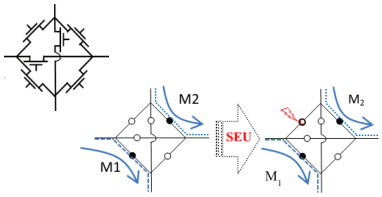

Figure 4: Basic switch structure and SEU effect [13]

In the past decade, various SEU mitigation techniques to detect, mask, or modify the fault effect for FPGAs have been proposed. According to the design, the designers have utilized a type of redundancy in the data layer, software layer, hardware layer, and over the time. In some cases, hybrid methods are used for this purpose. Triple modular redundancy (TMR) is a well-known hardware redundancy technique for masking SEU effect in integrated circuits. For FPGAs, this mitigation method can be applied in a single FPGA. However, it requires special consideration to overcome some pitfalls which are happening when TMR is used in a single FPGA. Due to the fact that all the logic paths and the flip flops are susceptible to SEU faults, full module redundancy is required in FPGAs.

To provide the triple modular redundancy, the designers should divide the scheme in three parts: logical parts, Flip-Flops, and input/outputs. For the combinational logic (time-independent logic), the scheme is triplicated and a majority voter like 0, is placed in the outputs. Also for the output/input pins, the internal 3-state buffers are used instead of Look-Up Tables (LUTs), which are used to implement all the Boolean functions in FPGA based designs as shown in 0.

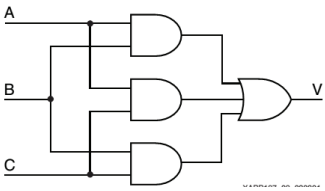

Figure 5: Majority Voter Circuit [16]

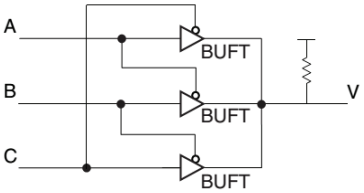

Figure 6: Output/Input Buffer Style Majority Vote Circuit [16]

For the sequential logic in which the Flip-Flops are used, more consideration is required. For these sections, firstly all Flip-Flops are triplicated, then the outputs of each set of the same Flip-Flops are voted through a local voter. Finally, the output of the local majority voters is spread over the scheme. In order to apply TMR over the designer's scheme, Xilinx Corporation has presented XTMR tools. But, this tool is allocated for space grade devices and cannot be considered as COTS. Therefore, we have developed a soft framework that is able to automatically complete its process. In the following section, we have detailed the design and hardening of the proposed CAN controller in SRAM based FPGA

The proposed CAN bus controller in this paper is completely based on FPGA. Only in the physical layer, we have used a transceiver from NXP Semiconductor. According to the CAN 2.0A protocol, the proposed CAN controller supports 11-bit ID in the frames.

As illustrated in Figure 7:, the CAN bus layers includes three main subsets. In Object layer, in which the message is filtered and message status is handled. In the transfer layer, the fault confinement, error detection, bit stuffing, bit rate, message framing, arbitration, and validation is completed. In the physical layer, the appropriate signal level and transmission medium have been prepared.

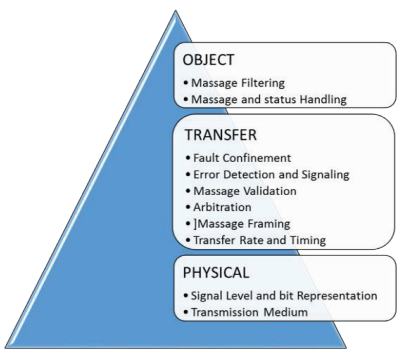

Figure 7: CAN Network layers

In the proposed scheme, we have completely covered the required parameters. According to the CAN protocol, in order to overcome the unwanted individual oscillations on CAN network, a bit is inserted every time five identical bits appears in line. As we know, the number of bits which are transmitted in a CAN network per second are represented by a parameter nominated as Bit Rate. Every node on CAN network must have the same predefined bit rate. In the proposed scheme, we have set the bus Bit Rate is equal to 250 kBit/s. Moreover, according to the CAN standard, four segments are defined for a nominated bit time that include: synchronization segment, propagation delay segment, and phase buffer segments 1 & 2. To determine the bit value, a single or multiple sampling is set in the middle of phase segment 1 and phase segment 2. We take three samples and vote among them. By the way, if two of the three are dominant, the sample is recorded as dominant.

The CAN network provides some Error detection mechanisms: bit check, frame check, CRC coding, Acknowledgement Check, and bit stuffing. According to these

mechanisms, the proposed scheme is able to detect an error in the data layer. Like all other controllers, after a bit is transmitted on the network, we read back and check the bit from line. If a collision is detected, this means that a node with higher priority is on network. Therefore, we stop data transfer and listen to line. Another fault detection mechanism is frame check, in which a collision is detected in some specific bit fields that always have fixed values. In this condition, we have provided an error frame on network. The CRC coding and check bits are another error detection mechanism. We always calculate the CRC codes for all received and transmitted data. A mismatch in CRC codes are treated as a faulty frame. Moreover, if an error condition is detected, the acknowledgement bit in the frame is handled to detect faults. As mentioned before, we have used bit stuffing mechanism in the proposed scheme. Therefore, if more than five identical bits in a data frame is detected, an error frame is transmitted. Finally, the fault confinement is a major and final step in the CAN protocol to provide a reliable data link [11]. We have used a generic parameter to limit the scope of fault affection into the local area, and protect other nodes from getting contaminated by a faulty node.

The finite state machine (FSM) of the proposed scheme is represented in 0. Each color in this figure illustrates a portion of the proposed controller tasks. The idle state or initial state is the zero state which has a gray color. In this state, the FSM waits for a frame sends request from top level, or expects a new frame from CAN bus. The required states to receive a frame from bus are distinguished by green color, and the red states represent the associated states to send a frame on bus. After a frame is formed in the second state, we may encounter with a priority collision in the third state. In this case, the FSM switches to state 1, and starts to receive a high priority message. It is required to mention that the error detection mechanisms are used in all states. As soon as, an error is detected, an error frame is transmitted. Fault management and handling start from state 6, when all the standards are completed. After any frame is transfered on CAN network, all nodes must release the bus. The blue states in FSM are used for this purpose. Moreover, the handshaking with the top level is completed in the orange states. After a transmission on the bus is completed, the FSM returns to its initial state.

When an error is detected, the state machine is switched to error frame transmission state. Also, when an overflow occurs on the bus, the overload frame is handled. In the proposed scheme, we initialize these conditions in state 6 and 15. For both cases, the state machine waits until the frame capture is completed (Rx = 0). Then, the controller transmits 6-12 zero bits on bus. Finally, seven recessive bits are set on the bus to terminate the frame transfer. 0 and 0 illustrate the required states to provide overload and error frames in the proposed scheme.

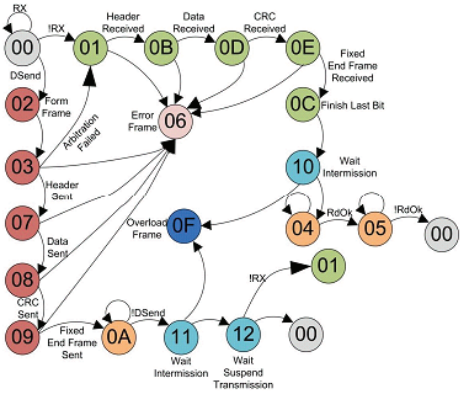

Figure 8: Main finite state machine

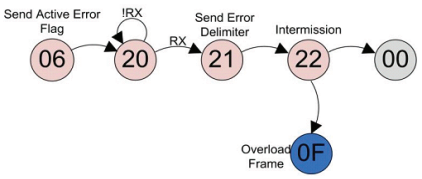

Figure 9: Finite state machine for error frame

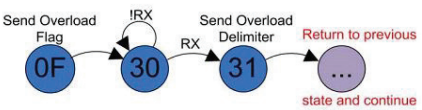

Figure 10: Finite state machine for overload frame

In order to mitigate the SEU faults, we have used these approaches in our designs: hardware duplicating over the standalone hardware CAN controller, duplex with comparison inside the FPGA, triple modular redundancy over the proposed HDL CAN controller in FPGA, and dual duplex with comparison inside the FPGA over the proposed HDL soft core as shown in 0 to Figure 15:. For the subsystems that have no FPGA, as illustrated in 0, we have duplicated the CAN standalone controller and all other related devices. For the FPGA based subsystems, we have investigated DwC, TMR, and DDwC versions of the proposed scheme. As mentioned, to triplicate the design, we have used three similar units with a voter. We have also cut out the outputs of all Flip-Flops as shown in 0, and fed back the output of the voter to each replica.

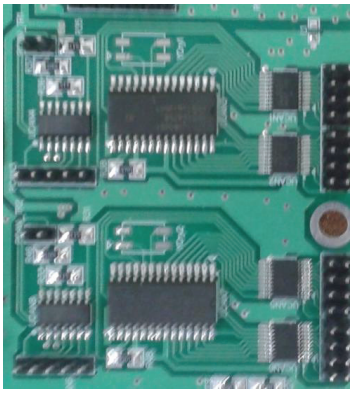

Figure 11: Full duplex over the standalone CAN controller on hardware

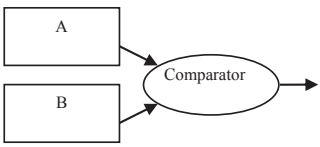

Figure 12: Full duplex with comprator (DwC) inside the FPGA

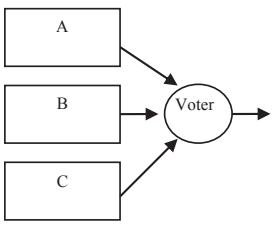

Figure 13: Full triple modular redundancy (TMR) inside the FPGA

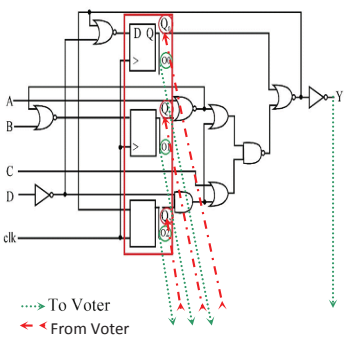

Figure 14: Special Triple modular redundancy in FPGA

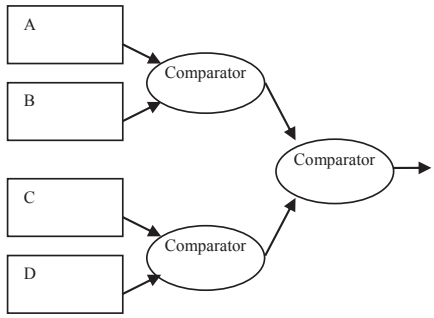

Figure 15: Dual duplex with comprator (DDwC) inside the FPGA

III. RELIABILITY ANALYSIS

The RAMS expression refers to Reliability, Availability, MTTF, and Security which are related to the ability of the system to achieve an acceptable level of reliability in communication satellites. A capability of the system component to continue their functionality under faulty condition is defined as reliability. Some external and internal factors may affect the reliability of an FPGA-based system. The internal factors refer to dissipation and defects that occur during the manufacturing process, while the external factors are mostly about the environmental issues like space radiations. All of these factors are summarized in a λ parameter which is defined as failure rate. In the previous section, we have introduced three basic structures to develop a fault tolerant FPGA-based CAN from the proposed plain version. In this section, we have analyzed the reliability of the aforementioned structures. First of all, it is better to mention that if the failure rate of a system is λ, its reliability is defined as R (t) = Exp (-λt) [17]. In order to calculate the reliability of the other structures, most literatures have provided Poisson Process and Markov Chain [17]. Based on these methods, for Duplex with comparison approach, we have:

$$R_{DwC}(t) = R_{com.}(t)\left[R^2(t) + 2CR(t)\left(1 - R(t)\right)\right] \quad (1)$$

In which, $R_{com.}(t)$ is defined as reliability of comparator. C is the Coverage Factor, which is defined as the probability that a faulty processor will be correctly diagnosed, identified and disconnected. The fault converge has a major role in the reliability of this method. The CAN protocol supports some internal fault detection mechanisms such as: bit stuffing, CRC check and so on. These mechanisms provide an acceptable level of coverage factor.

For triple modular redundancy, the reliability can be calculated according to (2). R voter (t) is defined as reliability of the voter.

$$R_{TMR}(t) = R_{\text{Voter}}(t)\left[3R^2(t) - 2R^3(t)\right] \quad (2)$$

For the last structure, DDwC, we have duplicated two DwC structures with a new comparator. If we assume the

coverage factor of the new structure is remained as the same, the reliability of this method can be calculated as:

$$R_{DDwC}(t) = R_{com.}(t)\left[R^2_{DwC}(t) + 2CR^2_{DwC}(t)\left(1 - R^2_{DwC}(t)\right)\right] \quad (3)$$

In order to have a more sensible parameter, the average time to failures with the modeling assumption that the failed system is not repaired is defined as the mean time to failure(MTTF), which is defined as: MTTF = $\int_0^\infty R(t)dt$. If the comparator and the voter have been assumed fault free, we have:

$$\begin{aligned} MTTF_{Plain} &= \frac{1}{\lambda} \\ MTTF_{DwC} &= \frac{1}{2\lambda} + \frac{C}{\lambda} = \frac{1+2C}{2\lambda} \\ MTTF_{TMR} &= \frac{3}{2\lambda} - \frac{2}{3\lambda} = \frac{5}{6\lambda} \\ MTTF_{DDwC} &= \frac{1}{\lambda}\left(\frac{(1-2C)^3}{4} + \frac{4C(1-2C)}{2} + \frac{4C(1-2C)^2}{3} + \frac{2C}{2} + 2C^2\right) \end{aligned} \quad (4)$$

According to these results, if we have an appropriate coverage factor, the MTTF for the dual duplex method is more attractive. Also, MTTF of TMR structure is less than MTTF of the plain version, and this is not desirable.

IV. Simulation and Implementation Results

The simulation and implementation results of the proposed design on industrial Xilinx Virtex-4 FPGA show that the minimum period of 6.792ns (Maximum Frequency: 147.224MHz) can be achieved by the plane version (0). Also, for the tripled version maximum frequency of 98.211MHz is obtained. Moreover, the device utilization for the plane and TMR versions are proposed in Table 1 and Table 2.

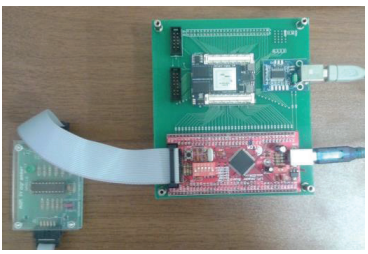

Figure 16: Test hardware structure

Table 1
Device utilization summary for plain version

Logic Utilization	Used	Available	Utilization
No. of Slices	469	5472	8%
No. of Slice Flip Flops	240	10944	2%
No. of 4 input LUTs	826	10944	7%
No. of bonded IOBs	64	240	26%
No. of BRAMs	2	32	6%
No. of GCLKs	469	5472	8%

Table 2
Device utilization summary for TMR version

Logic Utilization	Used	Available	Utilization
No. of Slices	1542	5472	28%
No. of Slice Flip Flops	723	10944	6.6%
No. of 4 input LUTs	2570	10944	27%
No. of bonded IOBs	64	240	26%
No. of BRAMs	6	32	18.7%
No. of GCLKs	469	5472	8%

After the design and hardening of the CAN controller scheme is complemented, we have provided appropriate simulation test-benches to test its function. Like previous works in this field, we attempted to perform all tests according to the CAN specifications which are detailed in [18]. In 0 and 0, data transmission simulation with and without bit stuffing have been represented. However, some specifications which require more challenges have been tested on the hardware platform through two completely the same platforms. A similar hardware is used to validate the priority issue in the CAN protocol. Furthermore, in order to insure that the proposed scheme is completely compatible with the other ASIC CAN controllers, we have tested the proposed scheme with AT90CAN32 which is a low-power CMOS 8-bit microcontroller that supports an embedded CAN controller Figure 19:. The simulation and hardware implementation results validate the correctness of the proposed scheme.

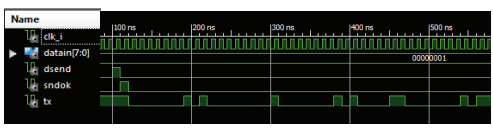

Figure 17: Simulation Results (without bit stuffing)

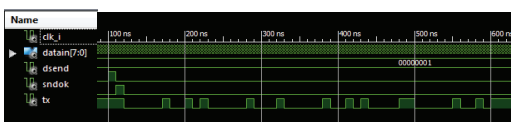

Figure 18: Simulation Results (with bit stuffing)

Figure 19: Entwicklungskit CAN2UART to test the proposed scheme

V. Fault Injection Platform and Results

After verifying the correctness of the designed and the implemented scheme by means of simulations and on real system, we have used DPR-FIP tool to inject single faults in fault tolerant versions to validate their capability in faulty conditions. The complete DPR-FIP platform is illustrated in [15]. The test setup, shown in 0 and 0, consists of a personal computer, fault injection controller, and FPGA platform. The personal computer provides a complete graphic user interface (GUI) to monitor and control of the SEU emulation process. The second part is a single event error (SEE) fault controller based on LPC2368 microcontroller. It receives the fault injection modes from PC and controls the configuration bit stream of FPGA. All required timings and signaling are managed through this external microcontroller. The third part is the FPGA platform which hosts the design under test (DUT) and other modules. The configuration space of the FPGA is divided in two separate segments. One part is allocated to DUT, which is in our case, it is CAN controller, and another one is assigned to the fault-free version of DUT, ICAP interface, a comparator, and other redundant versions.

Table 3 summarizes the resource utilizations and fault injection results. According to the results, the duplicated duplex with comparison have the best result in faulty condition, although it requires more resources. When the numbers of concurrent errors are increased, and the system is faced with multi bit upset faults, the mitigation capabilities of the dual duplex system are more outstanding. Table 4 represents the fault injection results in multi bit upset condition, in which we have investigated two and three bit upsets in the scheme.

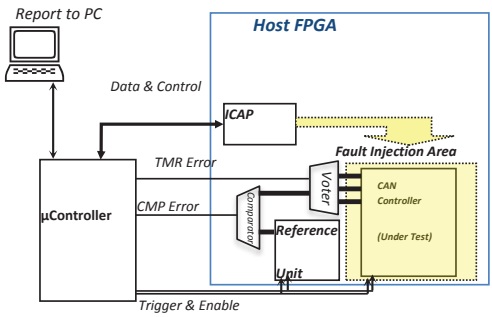

Figure 20: Block Diagram for SEU Fault injection

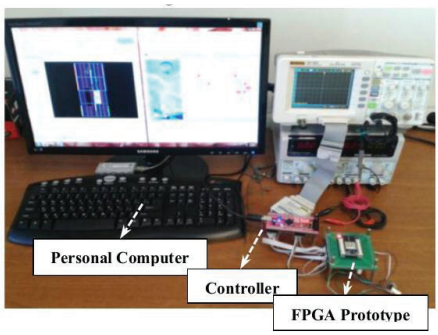

Figure 21: Hardware Set Up for SEU Fault injection

Table 3
Fault injection result in four version of CAN controller

Version	Source		SEU Fault injection results	
	Flip Flop	LUT tables	Switch Matrix	CLBs
Plain	240	723	231	481
DwC	488	1503	65	91
TMR	826	2570	23	86
DDwC	983	2988	8	12

Table 4
Multi Fault injection results in four version of CAN controller

Version	2-bit Upset Fault injection results		3-bit Upset Fault injection results	
	Switch Matrix	CLBs	Switch Matrix	CLBs
Plain	920	2344	3542	5432
DwC	1234	1982	2431	3214
TMR	910	1024	1294	1793
DDwC	154	192	213	482

VI. Conclusion

An implementation of a data handling protocol for aerospace applications can be achieved in different ways, but the FPGA based approach is more appropriate for deep space applications. This is due to the fact that the space qualified standalone CAN controllers and the embedded CAN microcontrollers are not developed appropriately. Continuous development in the FPGA technologies and the newly developed hardening techniques for FPGAs encourage the designers to propose a novel fault tolerated designs. In this paper, a fault tolerant data handling protocol based on FPGAs was presented. Through the simulation and hardware platform its capabilities were tested.

References

[1] Bisio, Igor; De Gaudenzi, Riccardo; Nguyen, Hung; Pavlidou, Fotini-Niovi; Yamazato, Takaya, "Recent advances in satellite and space communications", Communications and Networks, Journal of , vol.12, no.6, pp.523,528, Dec. 2010.

[2] Online: http://www.esa.int/Our_Activities/Space_Engineering_Technology/Onboard_Computer_and_Data_Handling/Onboard_Computer_and_Data_Handling2.

[3] Han-Way Huang, "Embedded System Design with C805", Cengage Learning, Standford, USA, 2009.

[4] M. Khurram and S.M.Y. Zaidi, "CAN as a spacecraft communication bus in LEO satellite mission", 2nd International Conference on Recent Advances in Space Technologies, pp.432,437, 9-11 June 2005.
[5] F. Bruhn, J. Köhler, L. Stenmark, "NanoSpace-1: The impacts of the first Swedish nanosatellite on spacecaft architecture and design", Acta Astronautica, Vol. 53, No. 4–10, PP: 633-643, 2003.
[6] M. Caramia, L. Bolognino and G. Furano, "Can Bus Solutions for Data Handling Space Systems" 5th Eucass - European Conference for Aerospace Sciences, Munich, Germany, EUCASS 2013.
[7] Online: www.can-cia.org/index.php?id=systemdesign-can-protocol
[8] D. Novak, A. Kerek, L.-O Norlin. G. Dajko, and Et al., "Memory irradiation measurements for the European SMART-1 spacecraft," 6th European Conference on Radiation and Its Effects on Components and Systems, Grenoble, France, pp 445-449, 10-14 Sept. 2001.
[9] Noel, P., "Mission Control Centre role in Thales Satcom systems," IEEE First AESS European Conference on Satellite Telecommunications (ESTEL), Rome, Italy, pp1-8, 2-5 Oct. 2012.
[10] European Cooperation for Space Standardization (ECSS), "CAN Bus extension protocol", ECSS-E-ST-50-15C DIR1, 13 December 2013.
[11] Aysan, H.; Dobrin, R.; Punnekkat, S., "Fault Tolerant Scheduling on Controller Area Network (CAN),"Object/Component/Service-Oriented Real-Time Distributed Computing Workshops (ISORCW), 2010 13th IEEE International Symposium on , vol., no., pp 226-232, 4-7 May 2010.
[12] T.R. Jena, A.K. Swain, K. Mahapatra, "A novel bit stuffing technique for Controller Area Network (CAN) protocol", International Conference on Advances in Energy Conversion Technologies (ICAECT), pp113-117, 23-25 Jan. 2014.
[13] Bruno Gaujal and Nicolas Navet "Fault Confinement Mechanisms on CAN : Analysis and Improvements", IEEE Transaction on vehicular technology, Vol. 54, No. 3, pp 1103–1113, 2005.
[14] R. Omidi Gosheblagh, K. Mohammadi, "Hybrid time and hardware redundancy to mitigate SEU effects on SRAM-FPGAs: Case study over the MicroLAN protocol", Microelectronics Journal, Vol. 45, No. 7, pp 870-879, 2014.
[15] R. Omidi Gosheblagh, K. Mohammadi, "Dynamic Partial based Single Event Upset (SEU) Injection Platform on FPGA", International Journal of Computer Applications, Vol. 76, No. 3, pp 19-24, 2013.
[16] C. Carmichael , "Triple module redundancy design techniques for Virtex FPGAs", Xilinx Apllication notes, XAPP197 (v1.0.1) July 6, 2006.
[17] I. Koren and C. M. Krishna. Fault Tolerant Systems. Morgan-Kaufman Publishers, San Francisco, CA, USA, 2007.
[18] R. Bosch, "CAN specification", Version 2.0, Motorola Corporation, 1991.