Trans. JSASS Aerospace Tech. Japan
Vol. 12, No. ists29, pp. Pj_15-Pj_24, 2014

Original Paper

Time and Space Redundancy Fault Tolerance Trade-offs for FPGA Based Single and Multicore Designs

By Mohamed Mahmoud IBRAHIM, Kenichi ASAMI and Mengu CHO

Kyushu Institute of Technology, Kitakyushu, Japan

(Received June 24th, 2013)

This paper investigates the gains and losses in terms of power, area, reliability, and speed when applying time redundancy fault tolerance techniques on single core designs compared to space redundancy fault tolerance techniques applied to multi-core designs. The system is developed on the virtex5 FPGA from Xilinx, it uses 65nm technology with a relatively moderate to high static power consumption. The system consists of two design alternatives. The first is a single core embedded processing system that applies time redundancy fault tolerance through execution repetition to perform self-check pointing through consensus. The second system is built from 3 soft IP core processors which perform a space redundancy approach through Triple-Modular-Redundancy (TMR) with feedback among the processors. The performance of both systems is evaluated in terms of the execution speed and latency due to fault tolerance techniques compared to the non-fault tolerant system.

Key Words: Multicore Systems, Fault Tolerance, Space Redundancy, Time Redundancy, FPGA

1. Introduction

Developing robust space avionics systems is a challenging task. As the missions diverse and increase their data processing requirements, the need for fast and reliable data processing systems emerges. Nevertheless, a balance should be hit between four main parameters: the power, the processing speed, the mass, and the reliability. It is required to optimize the design to have a reliable system with low power consumption and low mass while having high processing capabilities.

Nowadays, modern Field Programmable Gate Arrays (FPGA) provide the opportunity to develop complex digital designs with high speed and at moderate power consumption[1,2]. Single and multi-core processor systems are integrated with custom designed logic cores to serve the different design needs[3].

In developing space systems, several design techniques are commonly used to design reliable systems. Fault tolerance and fault avoidance are the common techniques[4-6]. Fault avoidance depends on preventing the faults from occurring in the functioning design. Fault tolerance depends on tolerating the effects that faults might introduce to the design in a way that keeps it functioning in an accepted performance. The concept of redundancy is the base for fault tolerant designs. Redundancy can take place in repeating the functioning design units, all or in part, with the same or diverse designs; in this case it is called space redundancy. A voter is used to judge between the results of the redundant units. Time redundancy is about repeating the execution of some of the program critical functions several times to reach a consensus among the results. Data redundancy is to add additional data bits to the original data where the additional bits will carry the Error Detection And Correction (EDAC) code that can be used to detect and correct faults in the data stream.

Software is a basic counterpart in developing complex system. Fault tolerant techniques are developed for software protection. N-version programming, N-copy programming, recovery blocks, and check-pointing are among the common techniques[7]. To protect the operation of a system, a hybrid of the techniques is used[8]. Fault injection and radiation testing are used to test the system robustness to bit flips

In this paper we present a comparative study between using single core processor in carrying the system tasks and using triple core redundancy. The comparison takes place in terms of power, speed, resources utilization and reliability. Both systems are implemented in a Static Random Access Memory (SRAM) based FPGA. The systems were designed using the Xilinx FPGA Virtex 5 LX50T. It is a 65 nm FPGA. The Embedded Development Kit (EDK) tool from Xilinx was used to design a single processor system and a triple processor system. Bubble sort algorithm was implemented and run on both systems to detect the average speed when applying redundancy in implementing the algorithm. The power consumption and resources utilization were estimated and the FPGA design reliability is calculated in both cases.

The objective of this work is to clearly understand the advantages and disadvantages of using space and time redundancy in 65 nm FPGAs. The trade-offs in selecting either of the two techniques are: selecting a design that is economic in its power consumption, provides high reliability, performs in a high standard, and achieves reasonable utilization of the FPGA resources.

The paper proceeds as follows: Section 2 presents the different architecture alternatives when developing an embedded processor system. The systems designs are presented in section 3. The reliability estimation procedure based on in-orbit investigation is presented in section 4. The results of testing the systems are presented and discussed in section 5. The conclusion and future work are presented in section 6.

2. Architecture Alternatives

When developing an embedded system, the main variant for the different architectures is about how the processor and memory are interfaced. The simple embedded processor system as shown in figure 1, consists of a system bus a microprocessor or microcontroller and other peripherals connected to the bus. The peripherals might contain timers, interrupt controllers, Input/Output processors, Direct Memory Access (DMA) controllers, and custom logic that implements specific functions. The memory and the system bus are crucial parts in the architecture when many processors are to be integrated together. They define how the processors access the memory for data storage and retrieval and/or instructions fetching.

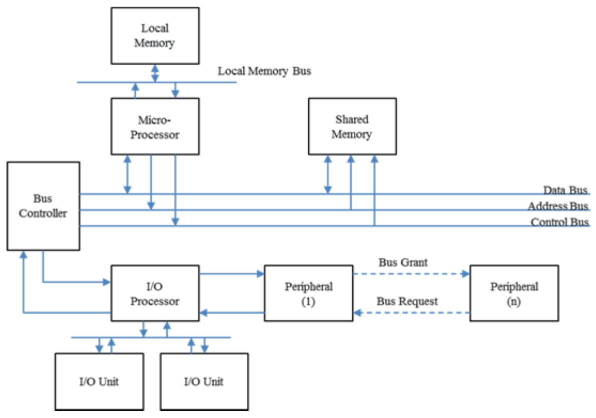

Fig. 1. Simple embedded system architecture.

In our previous paper[9] we have shown that four general architectures do exist when classifying the memory and processor interfaces:

1- Multi-Processor-Multi-Memory (MPMM)
2- Multi-Processor-Single-Memory (MPSM)
3- Single-Processor-Single Memory (SPSM)
4- Single-Processor-Multi-Memory (SPMM)

The difference among the four architectures is in the number of processors and memories which are interfaced together. The system reliability is affected with the chosen architecture. The calculation of the system reliability depends on the reliability of the attached memory as well as the processors. It is important to notice that single processor system whether connected to single memory or multiple memories is used when time redundancy in executing the software that runs on the processor will be adopted. In the case of the single memory the data can be stored in multiple buffers to provide redundancy in the storage. Multiple-Memory systems maintain an exact copy of all memory contents between the redundant units. The system can still have an additional form of redundancy by storing data in a redundant form in each memory while still having each memory repeated in a space redundancy.

The Multi-Processor systems have two conditions: the shared memory and the non-shared memory. In the shared memory systems the processors share the memories where they store and retrieve the data as well as the code memories from which they fetch the instructions. The MPSM and the MPMM with shared memories are examples of a tightly coupled multiprocessor system. The MPSM (non-shared) and the MPMM (non-shared) are examples for the loosely coupled multiprocessor systems.

The design of a fault tolerant embedded system usually merges different techniques together. The use of redundant memories and processors adds to the reliability as well as increasing the complexity of the system and its power consumption. A system that contains reasonable number of units in space redundancy, to maintain the power consumption and reliability, while adopting time redundancy techniques, is the ultimate choice.

In our designs we test the SPSM architecture and the MPSM (non-shared) architectures. The SPSM makes use of repetition of execution over the time and storage of results in extra copies. The MPSM makes use of the space redundancy concept where three processors operate in parallel to calculate the same operations to reach a consensus. Both systems run a bubble sort algorithm for comparing their performances. The test is run for 100 times and each time a vector of length 100 words is randomly generated. The time histogram for sorting the vector and performing the fault tolerance check of the results voting is plotted for the non-fault tolerant system, the space and the time redundancy systems.

3. System Design

Two separate systems were implemented to test the trade-offs of using time and space redundancy in the Virtex-5LX50T FPGA: single processor system and Multi-Processor system. In the single processor system as shown in figure 2, a MicroBlaze Processor is connected to a Local Memory Bus (LMB) where a local Block Random Access Memory (BRAM) is attached. The processor is connected to other peripherals through the Processor Local Bus (PLB). A watch dog timer is used to reset the system in case the processor stopped working. The processor receives an interrupt from the Interrupt Controller (INTC) that the watch dog timer finished counting and should be reinitialized. If the processor was working and did not hang up, it will respond to the watch dog timer interrupt and will reset it. In case the processor stopped working for any reason, it will not respond to the watch dog timer interrupt. The watch dog timer will send a reset request to the reset module. The reset module will then reset the whole system including the processor.

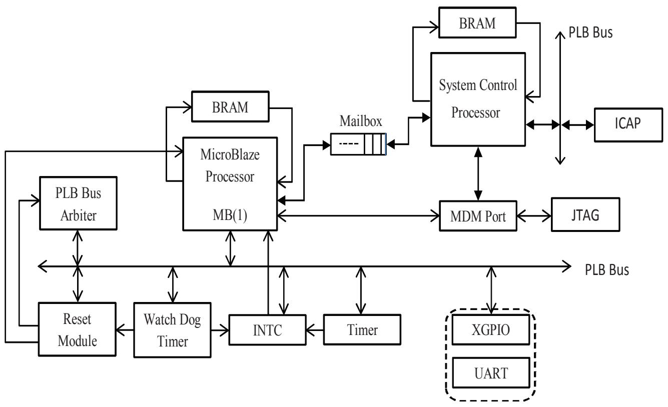

Fig. 2. Single processor system.

The system contains a system control processor which might handle the detection and correction of single bit errors that might happen in the configuration bit stream, check pointing of the current status, and restoring the system operation to the last known good status in case of reset. The communication between the system control processor and the Microblaze processor takes place through the mailbox IP core. A timer is included in the system to provide the ticks needed for an operating system to operate such as the Xilinx Kernel. Two Timers exist in the same timer IP core, one of them is used for measuring the execution time of the code in this experiment. The Microblaze Debug Module (MDM) port is used to debug the software application running on the Microblaze processor and the system control processor. The peripherals are connected to the PLB. Xilinx General Purpose Input Output (XGPIO) IP core is used to input and output digital signals. It provides a control interface to the outside world. The Universal Asynchronous Receiver Transmitter (UART) is used to send and receive serial streams to and from the processor.

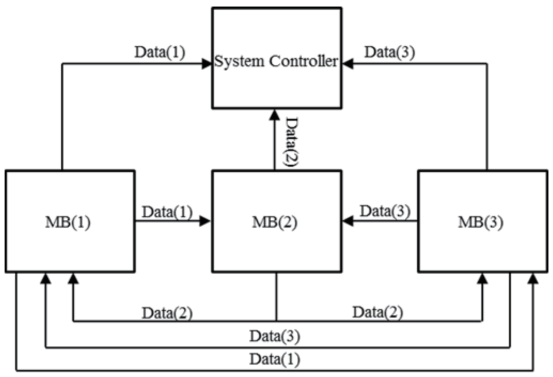

Fig. 3. Inter-processor communication mechanism.

When three processors are used, their data is exchanged among them after each execution cycle. They use the mailbox IP core as an inter-processor communication mechanism. As shown in figure 3, the three processors send the data to each other. Each processor would perform voting on its own locally generated data and the data provided by the other two processors. The voting takes place on bit level according to Eq. (1):

$$V = A.B + A.C + B.C \tag{1}$$

where A, B, C are binary words and the logic operations are the logical (AND) and logical (OR), (V) is the voting result. The results of the voting are then used by each processor in its operation. We call this technique, the inter-processor cross-voting through mailboxes. In our design the system controller keeps a copy of the last valid data from each processor in a Recovery-Block. Whenever the voting fails in two out of three processors, absence of consensus, the system controller performs a Roll-Back to the last stored good data in the Recovery-Block. The system controller keeps a score record for each processor that generates correct output. The output validity is judged at the system controller using acceptance tests. Whenever a processor generates a correct output its score is incremented by one. The data of the two processors with the highest scores can be used as reliable sources for the Roll-Back procedure.

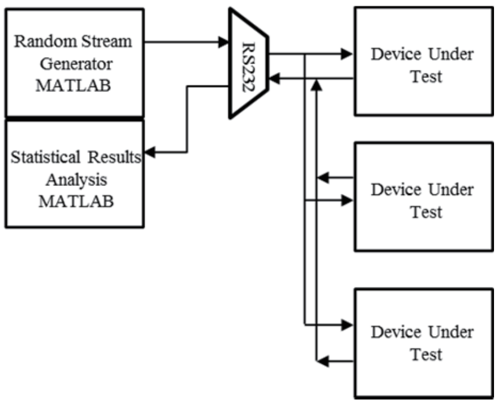
Random Stream Generator MATLAB
RS232
Device Under Test
Statistical Results Analysis MATLAB
Device Under Test
Device Under Test

Fig. 4. System operation concept.

The operation concept of the system is shown in figure 4, a random number stream of specified length, in this case it is 100 words, is generated by random number generation in MATLAB. It is then sent to the processors through the serial port RS232 interface. The processors save the received vector and wait for a signal to start the bubble sort algorithm. When the start signal is issued by the MATLAB script, the processors start their local timers to calculate their execution speeds and then initiate the Bubble sort algorithm execution. After the bubble sort finishes execution on each processor, the processors exchange the values of the sorted vectors among each other through the inter-processor mailboxes. Cross voting takes place at each processor. The local timer at each processor is stopped after the cross-voting and the execution time of each processor is sent to the MATLAB script. The test cycle continues until the required numbers of trials are executed. In case of single processor, it stores three copies of the data vector in its local memory. The bubble sort function runs three times and the sorted vectors are stored in three different memory locations. The voting takes place between the vectors stored in the local memory. The values of the execution times, after finishing the voting process, are sent from the processor to the MATLAB script. The execution times of each processor are statistically analyzed by the MATLAB script to calculate the mean and the variance of the execution times. Figure 5, shows the flowchart of the operation algorithm in case of single processor and triple processors.

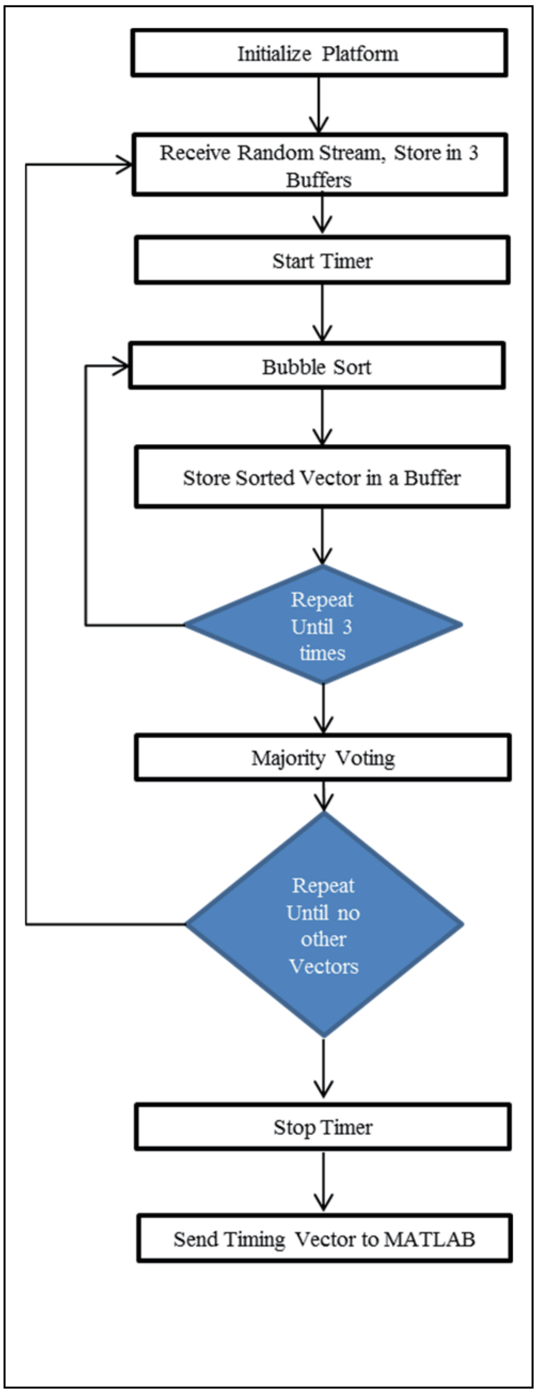
Initialize Platform
Receive Random Stream, Store in 3 Buffers
Start Timer
Bubble Sort
Store Sorted Vector in a Buffer
Repeat Until 3 times
Majority Voting
Repeat Until no other Vectors
Stop Timer
Send Timing Vector to MATLAB

Fig. 5a. Flow chart of operation in case of 1 processor system.

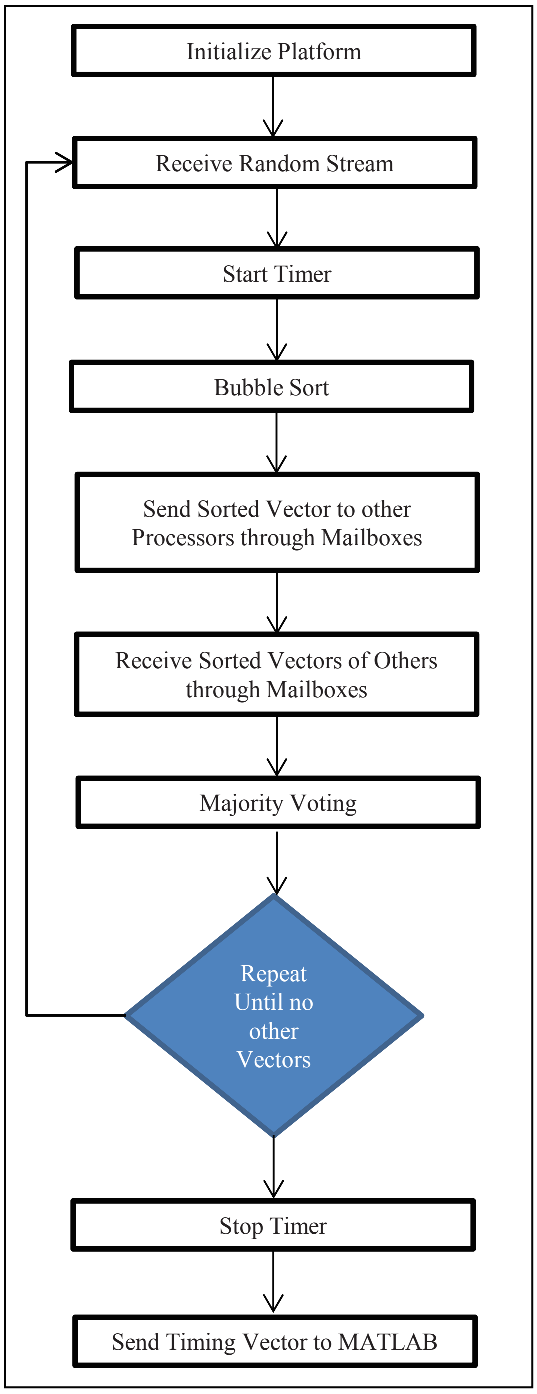
Initialize Platform
Receive Random Stream
Start Timer
Bubble Sort
Send Sorted Vector to other Processors through Mailboxes
Receive Sorted Vectors of Others through Mailboxes
Majority Voting
Repeat Until no other Vectors
Stop Timer
Send Timing Vector to MATLAB

Fig. 5b. Flow chart of operation in case of 3 processors system.

Fig. 5. Operation algorithm flow chart.

4. Reliability Estimation

Investigation of in-orbit SEU rates should be used when estimating the reliability of the design against soft errors in the configuration bitstream. SEU rates can be accurately investigated using accelerator radiation testing[10]. The Xilinx Virtex-5 XC5VLX50T FPGA consists of approximately 11.37 Mbits of configuration cells[11]. The device contains 355,190 configuration words and each word contains 32 bits (total of 11,366,080 bits). In order to approximately estimate the reliability of the design, in the worst case, we multiply the percentage of used resources by the device size in Mbits[11]. In calculating the effective configuration bits size of the XC5VLX50T FPGA device, only relevant configuration cells which would remain unchanged are included. No block memory contents are included in the calculation as their data might change during the design operation. Therefore the (effective configuration image size) is 11.37 Mbits without the block memory contents. The full configuration image size which contains the block memory contents is larger than 11.37 Mbits[11].

To estimate the expected reliability of the Virtex-5 XC5VLX50T device while operating in space we apply the following steps:

1- Define a target orbit upon which the estimations would take place. Use the target orbit data to calculate the trapped proton and electrons spectrum using the AP-8 and AE-8 models[12-14].
2- Use the static proton and heavy ion test results of the device, if they exist, or use the results of another device from the same technology, to estimate the (in-orbit SEU rates) using the Cosmic Ray Effects on Microelectronics (CREME-96) model within the Space Environment Information System (SPENVIS) online package[15-17].
3- Calculate the design specific SEU rate, soft error rate in design data, based on the design size in bits.
4- Calculate the system failure rate based on the design specific SEU rate. Assume that each SEU would cause a failure in the system operation. According to [11], this is a very pessimistic assumption as less than 20%, and typically less than 10%, of the configuration bits would cause design failures if they are subjected to upsets.
5- Reliability estimation based on the soft error failure rate (λ) and the time between scans of the FPGA configuration memory (T).

4.1. Target orbit definition

We select an orbit that is similar to the missions launched by Kyutech to demonstrate the estimation procedure of the expected system reliability[18]. Any other orbit can be selected depending on the target mission requirements. The orbit definition affects the expected charged particles fluxes. Table 1 shows the orbital parameters of the Horyu-2 satellite which is used as an example to carry out the reliability calculations.

Table 1. Horyu-2 satellite orbital parameters.

Orbit Type	general
Perigee Altitude	651 km
Apogee Altitude	671.6 km
Inclination	98.17°
R.asc of asc. node	223.04°
Argument of Perigee	31.95°
True Anomaly	328.25°

We use SPENVIS online package to calculate the fluxes of the trapped protons and electrons, solar particles, and galactic cosmic ray in the chosen mission orbit[17]. The output of the AE-8 and AP-8 models with solar maximum condition is shown in figures 6 and 7. It is clear that higher fluxes are focused at the South Atlantic region in what is known as the South Atlantic Anomaly (SAA). The spectra of the trapped electrons and protons energies versus the flux are shown in figures 8 and 9.

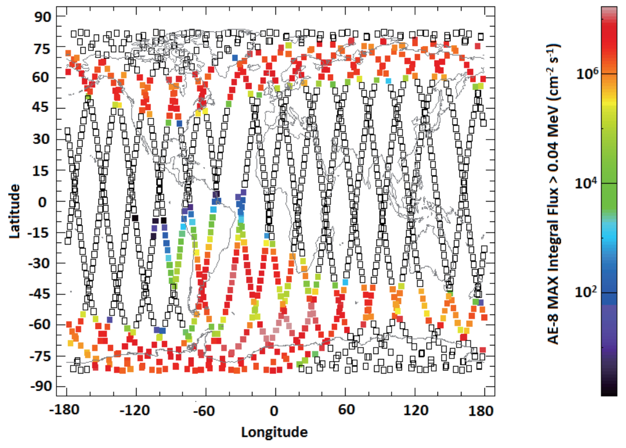

Fig. 6. AE-8 model trapped electrons flux in the orbit.

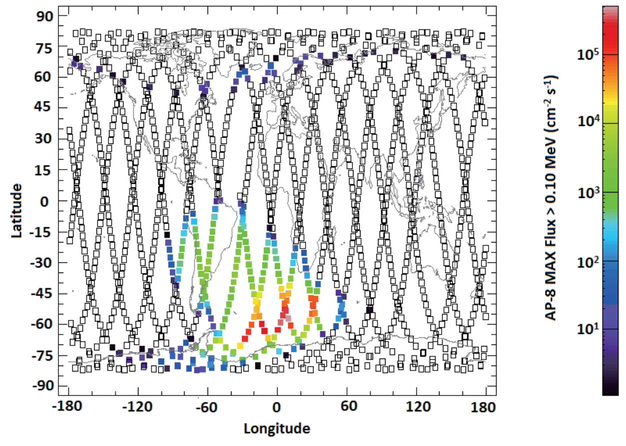

Fig. 7. AP-8 model trapped protons flux in the orbit.

The flux of the trapped electrons energy above 6 MeV is very low, less than 1 cm^{-2}sec^{-1}. The flux of the trapped protons energy at about 300 MeV is about 10 cm^{-2}sec^{-1}. Therefore, the expected effect of the trapped protons in causing bit upsets is higher than the effect of the trapped electrons. The threshold energy level at which the bit upsets are noticed depends on the specific device technology and can be found through radiation experimental results[19].

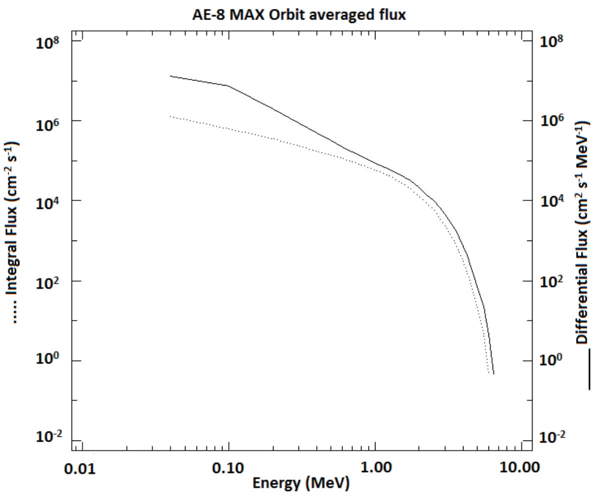

Fig. 8. Trapped electrons energy spectra versus the flux.

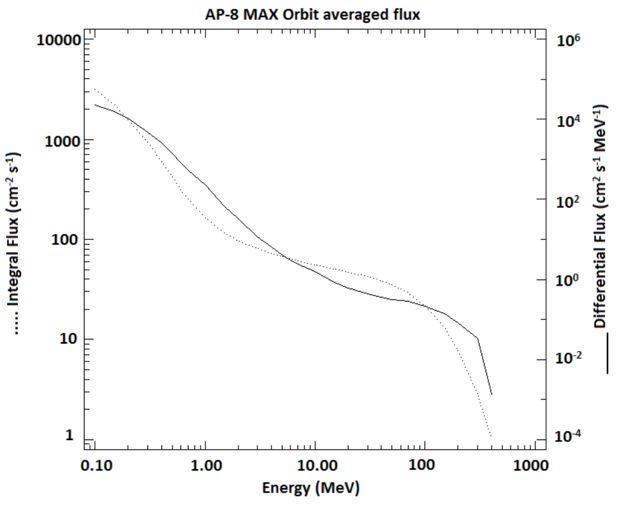

Fig. 9. Trapped protons energy spectra versus the flux.

4.2. SEU rate estimation

After defining the target orbit and estimating the trapped protons and trapped fluxes at different energies using the AP-8 and AE-8, we use the output to estimate the SEU rate using CREME-96 model in SPENVIS online package[15-16)]. When running the simulations, we assumed that the satellite has an Aluminum shielding thickness of 0.5 g/cm^2. We used the real experimental results of the static proton and heavy ion testing of the XC5VLX50 FPGA device as inputs to the CREME-96 model, in order to estimate the device specific upset rates[19)]. The system was built on the XC5VLXT50 FPGA device. However, both the XC5VLX50 and the XC5VLXT50 FPGA devices are identical in the number of slices except that the XC5VLXT50 FPGA contains some extra hard cores for advanced serial connectivity[20)]. The XC5VLX50 FPGA proton event bit cross-section is $8.61\text{x}10^{-14} \pm 4.90\text{x}10^{-16}$ cm^2/bit for 200 MeV and $6.37\text{x}10^{-14} \pm 1.17\text{x}10^{-15}$ cm^2/bit for 65 MeV[19)]. The heavy ion event bit cross-section test results were fit using weibull curve. It is calculated from the number of single ionized particle-induced events in the data set (i.e., each Multiple Bit Upset (MBU) counts as one event)[19)]. The four parameters of the weibull curve are the saturation cross-section or limit (σ_{lim}), the threshold or onset energy (E_o), the width of the rising portion of the curve (W) and the power that determines the shape of the curve (S) [17)]. From [19)] the values of the four parameters are (σ_{lim}) or (L=5.73E-8) cm^2/bit, (E_o) or (L_o=0.5) MeV-cm^2/mg, (W=15) MeV-cm^2/mg and (S=1.5). The heavy ion event bit cross-section for the highest tested LET of 68.3 MeV-cm^2/mg is $5.73\text{x}10^{-8}$cm^2/bit[19)]. The SEU rate along the first 14 orbits, about 23 hours of flight, is shown in figure 10. The peaks represent the rate at the SAA region. We calculate the worst case reliability based on the SEU rate at the SAA regions of the orbit (highest peak value corresponds to worst case) as well as the non-SAA regions (rest of the orbit). The highest SAA SEU rate is 3.73E-10bit^{-1}sec^{-1} and the non-SAA SEU rate is 2.07E-10 bit^{-1}sec^{-1}.

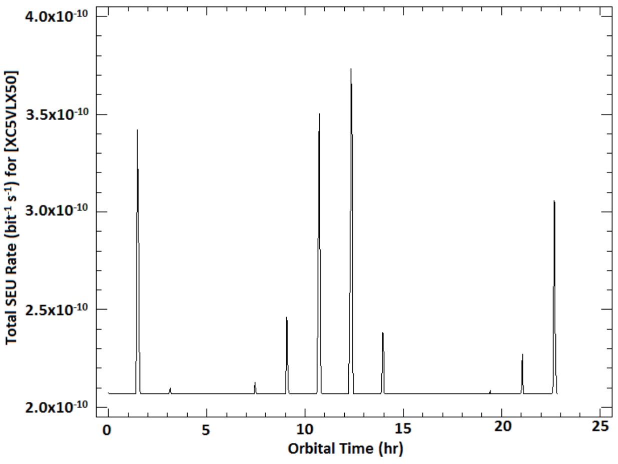

Fig. 10. In-orbit SEU rates estimated by CREME-96 and based on real radiation test results of the Xilinx XC5VLX50 FPGA.

4.3. Design specific SEU rate

The single core and multicore embedded systems slices utilization, design sizes and upset rates are shown in table 2. Each Microblaze system occupies 26% of the FPGA slices. The system control logic occupies 20% of the FPGA slices.

Table 2. Design size in bits and the upset rates.

		% of Occupied Slices	SEU rate at SAA sec^{-1}	SEU rate at non-SAA sec^{-1}
Single core	System control	20%	0.0008479	0.00047056
	Microblaze system	26%	0.0011023	0.00061172
	Total occupied slices	46%	0.0019502	0.00108228
Multi core	System control	20%	0.0008479	0.00047056
	3 Microblaze systems	3 x 26%	0.0033068	0.00183517
	Total occupied slices	98%	0.0041548	0.00230572

The design specific SEU rate is calculated by multiplying the in-orbit SEU rate (bit^{-1}sec^{-1}) at the SAA and non-SAA regions by the design size (bits) which is estimated from the (% of occupied slices x FPGA bitstream size (11.37 Mb)).

4.4. System failure rate

Assuming that each upset event would cause a system failure, which is a very pessimistic approach, the system failure rate per day can be estimated from the SEU upset rate per sec as in Eq. (2).

$$System\_Failure\_Rate\_per\_day = SEU\_RATE \times 86400 \tag{2}$$

Typically less than 20% of the design bits would be sensitive to the design operation[11]. Table 3 shows the system failure rate calculated for the system control processor and each Microblaze processor system at the SAA and non-SAA regions. The failure rates are identical for the system control logic and the Microblaze processor system in the single and multicore designs as they occupy the same size of the bitstream in both designs. Calculations are carried out for two cases: when considering that any bit upset would cause a failure and when assuming that only 20% of the bit upsets would cause a failure.

Table 3. System Failure Rates.

	Failure rate at SAA day^{-1}	(20% Sensitive Bits) Failure rate at SAA day^{-1}	Failure rate at non-SAA day^{-1}	(20% Sensitive Bits) Failure rate at non-SAA day^{-1}
System control	73.26	14.65	40.66	8.13
Microblaze system	95.24	19.05	52.85	10.57

4.5. Reliability estimation

Configuration memory scanning is performed to make sure that the memory which contains the design bitstream is free of errors. Configuration frames are readback through the Internal Configuration Access Port (ICAP). Each frame has 12 bits of Error Correcting Code (ECC) which is capable of Single Error Correction and Double Error Detection (SECDED). The frame data is checked through an ECC calculator to detect if an error exists or not. In case a single bit error exists the error syndrome would locate the error position. The error can be corrected by toggling the bit value at the position identified by the syndrome. If double errors or more exist in a single frame (i.e., MBU) then the error cannot be corrected. The FPGA should be reset if the bit is sensitive to the design[11,21-23]. Xilinx also provides a Cyclic Redundancy Check (CRC) primitive to check the integrity of the whole bitstream image as well as an SEU controller IP core for bitstream scrubbing. The Scrubbing process consists of (scanning) reading back the frames, calculating the ECC, correcting single bit errors, and writing back the correct frames. The scan period is calculated from Eq. (3).

$$Scan\_Period = Total\_Configuration\_Words \times {}^{1}\!/_{Clk\_freq} \tag{3}$$

The maximum scanning rate per day is calculated from Eq. (4).

$$Max\_Scan\_Rate\_per\_day = {}^{86400}\!/_{Scan\_Period(\text{sec})} \tag{4}$$

The XC5VLX50T FPGA contains 355,190 configuration words. When operating at 50 MHz scanning clock then the scanning period according to Eq. (3) is (scan_period = 7.1038 msec). The maximum scan rate per day is 12,162,504.58 scans. This rate is for continuous scanning where a new scan is started just after the previous scan finishes. According to Xilinx, the scanning rate is recommended to be at least 10 times the upset rate[22]. As scrubbing takes place, we assume the maximum lifetime of any failure to be equal to the time between complete scans, call it the scan cycle time. The reason for this assumption is due to the fact that the errors in the configuration stream would be discovered during the scanning process and corrective action would be taken, either by resetting the FPGA or correcting the upset if possible. Therefore the lifetime of any error is limited by the scan cycle time. The basic formula to estimate the reliability is shown in Eq. (5).

$$R = e^{-\lambda T} \tag{5}$$

Where (λ) is the soft error failure rate in unit of time and (T) is the maximum soft error lifetime which is equal to the time between scans. The reliability of the single core system is estimated based on Eq. (6).

$$R_{single\_core} = R_{system\_control\_logic} \times R_{Microblaze\_system} \tag{6}$$

The reliability of the multicore system is estimated from Eq. (7). and Eq. (8).

$$R_{multicore\_system} = R_{system\_control\_logic} \times R_{TMR\_Microblaze} \tag{7}$$

$$R_{TMR\_Microblaze} = 3R^2_{Microblaze\_system} - 2R^3_{Microbalze\_system} \tag{8}$$

From Table 3, the highest failure rates are for the system control logic and the Microblaze system at SAA which are 73.26 and 95.24 failures/day respectively. These failure rates correspond to the worst case reliability estimation. The lowest failure rates for the system control logic and the Microblaze system at the non-SAA region, when only 20% of the design bits are considered to be sensitive for bit upsets, are 8.13 and 10.57 failures/day respectively. These failure rates correspond to the best case reliability estimation. In figures 11, 12 and 13, we plot the reliability curve of the system at the worst case estimations.

The configuration memory scanning rates are selectable and can be designed to achieve the required system soft error reliability. If we use the SEU controller IP core from Xilinx [23], then we would have continuous scanning, thus the time between scans would be 7.1038 msec as calculated from Eq. (3). At this scanning rate the reliability of the single core and multicore systems is equal to 0.999999 as shown in figure 13. If a custom configuration stream scanning technique was used through the Internal Configuration Access Port (ICAP) or the external SelectMAP interface then we can estimate the reliability of the single and multicore systems based on figure 12. For example when the time between scans equals 20 sec, then the single core system reliability is about 0.96 while the multicore system reliability is about 0.98. It is important to notice that the multicore TMR system provides better reliability than the single core system till a certain lifetime (time between scans) , as shown in figure 11 which is equal to about 606 sec in this design. After that time the single core reliability becomes worse than the multicore TMR reliability.

Therefore, it is always desired to select the time between scans to be lower than that value.

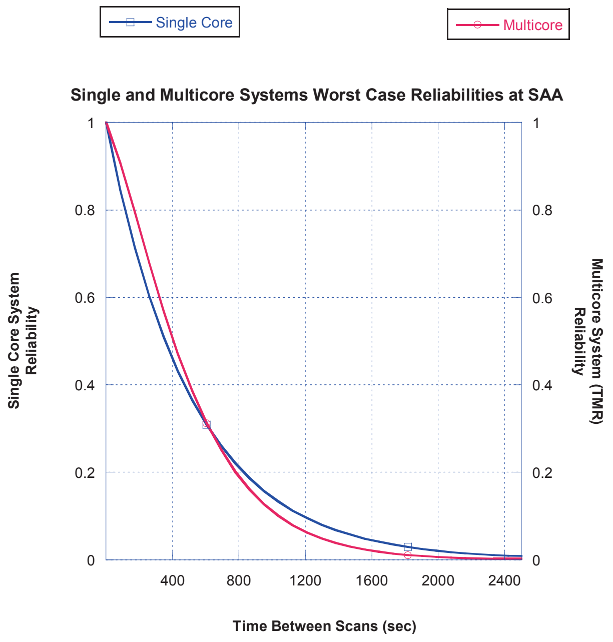

Fig. 11. Single versus triple processor systems soft-error reliability based on in-orbit estimation.

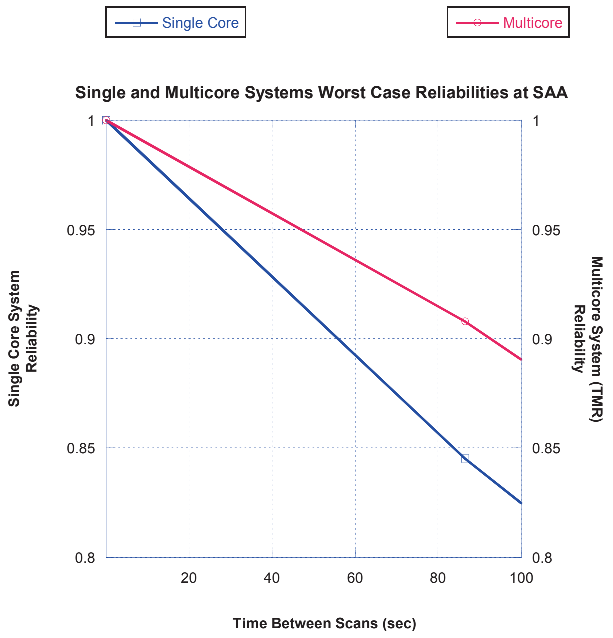

Fig. 12. Single versus Triple processor systems soft-error reliability based on in-orbit estimation, Time between scans ≤ 100 sec.

If an external MRAM is used for storing code or as platform configuration flash then the overall system reliability is estimated from Eq. (9).

$$R_{System\_soft\_error} = R_{MRAM\_soft\_error} \times R_{FPGA\_Design\_soft\_error} \qquad (9)$$

Where, $R_{MRAM\_soft\_rate}$ is the external MRAM memory soft error reliability while $R_{FPGA\_Design\_soft\_error}$ is the soft error reliability of the FPGA digital design. From the previous analysis, it is clear that increasing the scrubbing rate of the FPGA configuration stream leads to higher system reliabilities. Also, we noticed that the TMR multicore system provides better reliabilities than the single core system, in case non-continuous scanning is used, as long as it is below a certain value for the time between scans. As an example, if we assume the time between scans to be 20 sec, and MRAM soft error rate to be 0.05 failure/year then we can plot the total TMR system reliability as in figure 14.

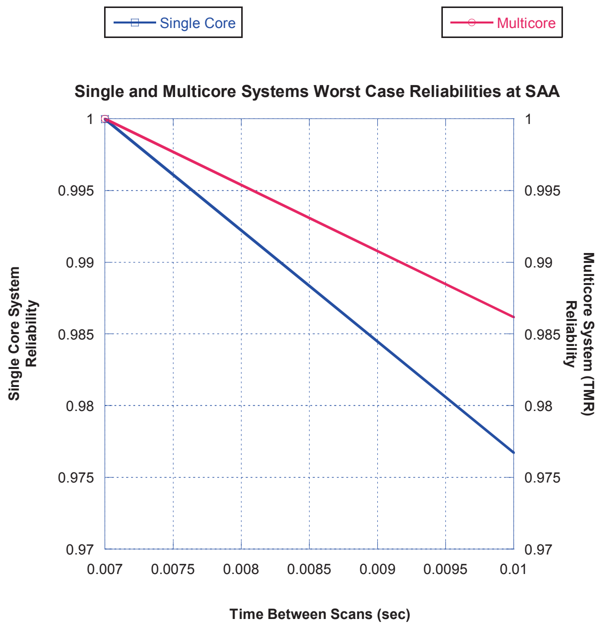

Fig. 13. Single versus Triple processor systems soft-error reliability based on in-orbit estimation, Time between scans ≤ 10 msec.

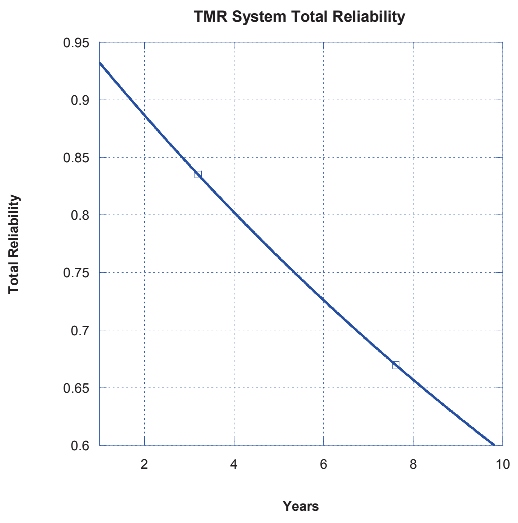

Fig. 14. Total System Reliability of TMR system based on in-orbit investigation of the FPGA SEU rates and assuming an MRAM soft failure rate of 0.05 failures/year.

5. Results and Discussion

The systems were tested using vectors of randomly generated data words. The time span of execution was collected for the non-fault tolerant system, the time redundant system and the space redundant system. Figure 15 shows the execution times histograms of the three cases. The execution times data were collected and grouped to depict their statistical distributions. We made curve fitting on the grouped data of each tested case using normal distribution. The graphs in Fig. 15 show the red line of the normal distribution fitting and the blue bars of the grouped data histograms.

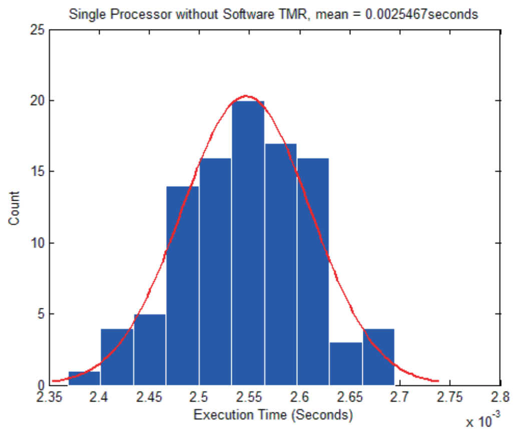

Fig. 15a. Non-fault-tolerance execution time histogram.

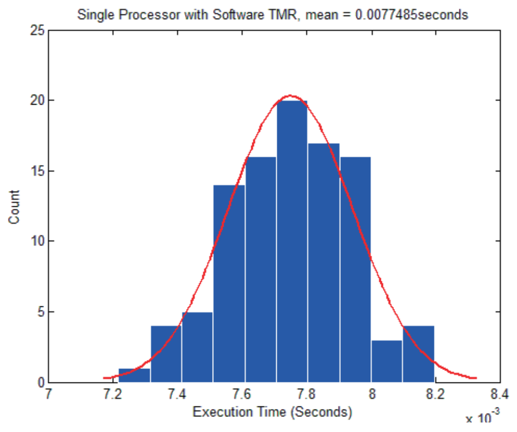

Fig. 15b. Time-redundancy execution time histogram.

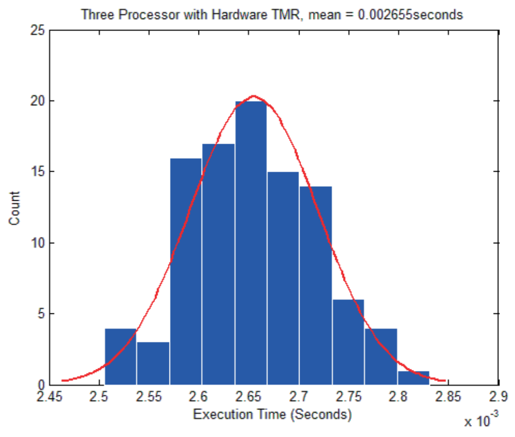

Fig. 15c. Space-redundancy execution time histogram.

Fig. 15. Execution times histograms for single and triple systems.

Each time a random sequence vector was generated based on the clock seed of the computer system running MATLAB to avoid repeating patterns of random sequences. The length of each vector was 100 words. The numbers of vectors applied during the test were 100 vectors. The statistical parameters of the normal distribution fitting of each histogram are shown in Table 4.

Table 4. Statistical analysis of execution times histograms in seconds.

Parameter	No-Fault Tolerance	Time Redundancy	Space Redundancy
Min	0.002354	0.007171	0.002463
Max	0.002739	0.008326	0.002847
Mean	0.002547	0.007749	0.002655
Median	0.002547	0.007749	0.002655
Mode	0.002354	0.007171	0.002463
Std	0.0001129	0.0003386	0.0001127
Range	0.0003852	0.001156	0.0003847

The mean execution time for the single processor without software TMR is about 2.55 ms. It is almost the same as the mean execution time of the hardware TMR, space redundancy, which is 2.65 ms. The mean execution time of the single processor with software TMR, time redundancy, is 7.75 ms. Almost 3 times higher than the non-fault tolerant and the space redundancy fault tolerance. This means that hardware redundancy is better in terms of execution time as it is as fast as the non-fault tolerant system which contains no overheads. However, we have to carefully notice that adding complicated data exchange protocols between the processors will add an execution overhead hence increasing the execution time significantly. If too much time is spent in handling the communication between the processors then the execution time might be close to the time redundancy case.

The power consumption varies between the single core and the multicore systems as shown in Table 5. The power consumption was estimated using the XPower Analyzer Tool from the Xilinx-ISE toolset. The total power consumed by the multi-core system is about 1.26 Watt which is only about 28% higher than the power consumed by the single core system.

The total power consumption consists from: 1) dynamic power, and 2) static power. The dynamic power consumption is affected by the size of the design and its operating frequency. The static power consumption is mainly affected by the transistor feature size of the FPGA and the operating voltage. The static power consumption in the FPGA transistors is considered as leakage. The leakage current and hence the leakage power increase as the feature size of the FPGA transistors is minimized. This leakage power is consumed even if the FPGA is not configured with any bitstream[24)]. About 0.46 Watt is dissipated in the form of leakage power in the multicore and single core designs. Table 5 represents the dynamic power consumed by the design resources: The on-chip clock trees, the logic gates, the design signals, the Block Random Access Memories (BRAMs), the Digital Signal Processors (DSps) which has a value of zero as we did not use DSPs in the design, the Phase Locked Loops (PLLs) and Digital Clock Managers (DCMs) for generating the required operating frequency and the Input/Outputs (IOs) logic. The difference in the dynamic power consumption between the two designs is about 0.278 Watt. This difference is due to the use of more cores, which means the use of more logic, signals, BRAMs, IOs and mostly more clock-trees, in implementing the multicore system. From table 5 we conclude that dynamic power consumption is responsible for the difference in the total power consumption.

Table 5. Power consumption in single and multi-core systems.

On-Chip	Single Core Power (W)	Multi-core Power (W)
Clocks	0.170	0.347
Logic	0.003	0.007
Signals	0.005	0.014
BRAMs	0.010	0.096
DSPs	0.000	0.000
PLLs	0.263	0.263
DCMs	0.068	0.068
IOs	0.001	0.002
Leakage	0.458	0.460
Total	0.978	1.257

The FPGA resources are almost utilized by the multi-core system while about less than one-third is utilized by the single core system as shown in Tables 6 and 7.

Table 6. Resources utilization in single core system.

Slice Logic Utilization	Used	Available	Utilization
No. of Slice Registers	5,932	28,800	20%
No. of Slice LUTs	6,866	28,800	23%
No. of Occupied Slices	3,337	7,200	46%
Number of BRAM/FIFO	18	60	30%
Total Memory Used (KB)	648	2,160	30%

Table 7. Resources utilization in multi-core system.

Slice Logic Utilization	Used	Available	Utilization
No. of Slice Registers	16,362	28,800	56%
No. of Slice LUTs	18,763	28,800	65%
No. of Occupied Slices	7,076	7,200	98%
Number of BRAM/FIFO	42	60	70%
Total Memory Used (KB)	1,512	2,160	70%

6. Conclusion and Future Work

This paper studied the effects of using hardware redundancy and software redundancy on the resources utilization, power consumption, execution speeds and reliability in single and multi-core designs of the Virtex-5 FPGA. The multi-core system makes better use of the resources and it executes at almost triple the speed of the single core system while its power consumption is only 28% higher than it. Thus, the multi-core system which uses space redundancy for implementing fault tolerance in the 65nm Virtex 5 FPGA through repeating redundant hardware processor cores is more efficient and effective than the single core design. In terms of reliability the TMR design is better than the single core design as long as the time between scans is kept below a specific threshold which is 606 sec for this design.

It is recommended according to the obtained results to make use of space redundancy approaches when designing digital systems using the Xilinx Virtex5 (65 nm) FPGA. This is valid due to the fact that considerable portion of the power consumed is dissipated in the form of leakage power. Adding extra logic did not add much to the total power consumed. The reliability of the space redundant system is higher or at least equal to the time redundant system below the critical threshold of the time between scans, and its execution speed is better as far as the communication protocol between the cores does not add much overhead. Resources are better to be utilized in the FPGA device rather than wasting them. The space redundant system makes higher utilization of the FPGA resources. We recommend repeating the work in this paper on different algorithms and applying a more time consuming communication protocol. A comparative study between the 65nm FPGA and other families such as the 28nm Virtex7 would be beneficial as well.

References

1) Xilinx website: http://www.xilinx.com
2) Actel website: http://www.actel.com
3) Asokan, V.: Designing Multiprocessor Systems in Platform Studio, Xilinx White Paper, WP262, **262** (2007), pp. 1-18.
4) Koren, I. and Krishna, M.: Fault Tolerant systems, ElSEVIER 2007.
5) Shirvani, P. P.: Fault-Tolerant Computing for Radiation Environments, Ph.D thesis, Center for reliable Computing, Stanford University, June, 2001.
6) Lima, F.D. and Carro, L.: Fault Tolerance Techniques for SRAM-based FPGAs, Springer, 2006.
7) Pullum, L. L.: Software Fault Tolerance Techniques and Implementation, Artech House, London, 2001.
8) Goloubeva, O.: Software Implemented Hardware Fault Tolerance, Springer 2006.
9) Ibrahim, M., Asami, K. and Cho, M.: Fault Tolerant Architecture Alternatives for Developing Nano-Satellites Embedded Computers, Online Proc. of AIAA SPACE 2012 Conference & Exposition, Pasadena, California, USA, September 11-13, 2012.
10) George, J. Koga, R. Swift, G. Allen, G. Carmichael, C. and Tseng, C . W.: Single Event Upsets in Xilinx Virtex-4 FPGA Devices, 2006 IEEE Radiation Effects Data Workshop.
11) Chapman, K.: SEU Strategies for Virtex-5 Devices, XAPP864 (v2.0), April 1, 2010.
12) Sawyer, D. M. and Vette, J. I.: AP-8 Trapped Proton Environment for Solar Maximum and Solar Minimum, NSSDC/WDC-A-R&S 76-06, 1976.
13) Vette, J. I.: The AE-8 Trapped Electron Model Environment, NSSDC/WDC-A-R&S 91-24, 1991a.
14) Vette, J. I.: The NASA/National Space Science Data Center Trapped Radiation Environment Model Program (1964-1991), NSSDC/WDC-A-R&S 91-29, 1991b.
15) Adams, J. H., Jr.: Cosmic Ray Effects on MicroElectronics, Part IV, NRL Memorandum Report 5901, 1986.
16) Tylka, A.J. et al.,"CREME96: A Revision of the Cosmic Ray Effects on Micro-Electronics Code, IEEE Transactions on Nuclear Science, **44** (1997), 2150-1260.
17) SPENVIS - online package website: http://www.spenvis.oma.be
18) Horyu-2 satellite website: http://kitsat.ele.kyutech.ac.jp/index_e_new.html
19) Quinn, H., Morgan, K., Graham, P., Krone, J. and Caffrey, M.: Static Proton and Heavy Ion Testing of the Xilinx Virtex-5 Device, Radiation Effects Data Workshop, 2007 IEEE, pp.177-184, 23-27 July 2007.
20) Virtex5 Family Overview: DS100 (V5.0), February 6, 2009.
21) Xilinx: Virtex-5 FPGA Configuration User Guide, Xilinx UG191 (v3.10), 2011.
22) Carmichael, C., Caffrey, M. and Salazar, A.: Correcting Single-Event Upsets through Virtex Partial Configuration, XILINX, XAPP216, (v1.0), 2000.
23) Chapman, K.: New Generation Virtex-5 SEU Controller, Xilinx, Version A.2 – s4th November 2009.
24) Curd, D.: Power Consumption in 65 nm FPGAs, Xilinx Corporation, WP246, **246** (2007).