A Fault-Tolerance Strategy for an FPGA-based Multi-Stage Interconnection Network in a Multi-Sensor System for Space Application*

Monica Alderighi, Fabio Casini, Sergio D'Angelo,
Davide Salvi and Giacomo R. Sechi

Istituto di Fisica Cosmica "G. Occhialini", CNR, via Bassini 15, 20133 Milano, Italy.
{monica, sergio, giacomo}@ifctr.mi.cnr.it

Abstract

Space research requires increasingly huge amount of scientific data. Next generation satellites will have on-board supercomputing capabilities to perform efficient information processing and overcome the possible limit imposed by communication bandwidth to ground receiving stations. They will also have to survive to even longer term mission, with respect to nowadays systems, thus reliability and fault tolerance will be a major concern, to cope with radiation induced faults. Flexibility also emerges as a desirable requisite for on-board processing system to implement new functionalities and run different algorithms for the ongoing mission. The trend is toward multiprocessor architecture in which processing nodes and memories are connected through high bandwidth interconnection networks. The paper presents a fault-tolerance strategy for an FPGA implementation of a redundant multistage interconnection network (MIN), for a space multi-sensor system. The implemented mechanism is endowed with careful fault diagnosis ability which allows to exploit MIN intrinsic reconfiguration capabilities, as well as the reprogrammability of SRAM-based FPGAs.

1. Introduction

Space research requires increasingly huge amount of scientific data, that demands for more and more complex and powerful computer systems to extract meaningful information from acquired data. Differently from traditional spacecraft architecture which only collect, package, and transmit data from multiple on-board instruments, next generation systems will have supercomputing capabilities to perform efficient on-board information processing and overcome the possible limit imposed by communication bandwidth to ground receiving stations [1],[2],[3],[4]. They will also have to survive to even longer term mission, with respect to nowadays systems, thus reliability and fault tolerance will be a major concern, to cope with radiation induced faults [5]. Moreover, flexibility emerges as a desirable requisite for on-board processing system, since it may offer several advantages at limited design cost increase. Flexible systems allow, in principle, to change or upgrade algorithms, depending on possibly changed environmental and/or faulty conditions, thus giving scientists the possibility of developing applications in their laboratory and then running them on the remote system with minimal or no re-engineering costs.

FPGA technology seems appropriate to implement high–performance processing systems. SRAM-based FPGAs have gained a growing interest in the space community due to their high gate density, low development cost, good resistance to high level of total ionising dose, and

* This work was supported by Agenzia Spaziale Italiana (the Italian Space Agency).

Proceedings of the 2001 IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT'01)
1063-6722/01 $17.00 © 2001 **IEEE**

IEEE COMPUTER SOCIETY

Exhibit 1057
Xilinx v. Sentient
IPR2023-00195

almost complete latch-up immunity. Besides, their unlimited reconfiguration capability allows to implement custom hardware yet maintaining the flexibility of software solutions. FPGAs offer numerous advantages such as: to perform on-board a great part of the processing tasks, that are normally implemented by ground-based software, with the possibility of reconfiguring the system from the earth; to implement new functionalities for the ongoing mission; to change data collection strategy in response to the quality of the received data or for changing instrumentation planning policy; to update the algorithms in order to adapt the system to changed operation conditions, advances in the astrophysics problem knowledge, and system faults.

Architectural trend for on-board processing system is towards parallel supercomputers made by a number of processing nodes interconnected by a high-speed, multiply redundant communication network.

A multi-sensor system to be used in space applications, such as future experiments on board the International Space Station, has been proposed [6]. It comprises a number of instruments, each one collecting data from a set of sensors. Instruments package data into data packets consisting, as usual, of a header followed by a data body. Data packets move to a number of memory elements through an interconnection network which is endowed with a control unit aimed at associating instruments to memory elements, depending on resource availability and scheduling policy. Memory elements are actually memory processors that convert a simple random access memory into a content addressable memory. This allows processing elements to retrieve data in a quiet easy way. Processed data are then sent to Earth by means of telemetry resources. The system architecture, as well as its operating system are under development at our institute.

This paper concerns the FPGA implementation of a redundant multistage interconnection network (MIN) for our multi-sensor system. The implementation has been accomplished by means of Xlinx 4000 family components. These are SRAM-based devices and, as such, are prone to radiation induced faults, which are mostly Single Event Upsets (SEU) or bit-flips in configuration or user memory cells.

In the paper a tolerance strategy of radiation induced faults for the implemented MIN is presented, based on line test and reconfiguration; both MIN and FPGA reconfiguration capabilities are exploited.

Some papers can be found in the literature reporting different reconfiguration-based techniques that face with faults affecting FPGAs. Roy and Nag [7] proposed a technique which made use of improved interconnection resources within the device to allow routability in the presence of faults; their technique was developed for antifuse-based FPGAs. Hanchek and Dutt [8] developed a technique for yield improvement that needed device architecture modifications and the adoption of specific place-and-route tools. Lach et al. [9] proposed a tiling approach that used one spare block per tile; such technique was claimed to have low overhead but, unfortunately, no information was given about how faults were detected within the device. Emmert et al. [10] presented a detailed review of related works in these fields, and proposed a fault tolerance technique for partially reconfigurable devices. Their technique made use of two (one horizontal and one vertical) self-testing areas (STARs) that rove continuously along the array of programmable blocks, thus performing exhaustive test on both used and spare logic and interconnects. The major drawback of this technique is the excessive fault latency: whenever a fault occurs within the working area, in a location which is far from the current position of the STARs, then wrong result could propagate through the system for long. So, while the STAR approach provides an effective way to perform concurrent fault diagnosis, it is not adequate as a fault tolerance technique for highly reliable applications.

Proceedings of the 2001 IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT'01)
1063-6722/01 $17.00 © 2001 **IEEE**

IEEE COMPUTER SOCIETY

The technique proposed in the paper makes use of self checking circuitry for on line fault detection and of a specific mapping of the self checking MIN onto the FPGA that permits an accurate diagnosis, driving the selection of the most appropriate repair procedure.

Section 2 sketches the interconnection structure of the multi-sensor system. The characteristics of the FPGA implementation of the MIN are treated in Section 3. Section 4 illustrates the fault-tolerance strategy, while results about MIN fault-tolerance capability are reported in Section 5. Finally conclusions are drawn in Section 6.

2. Interconnection Network

Many interconnection networks have been proposed in literature, among them Multistage Interconnection Networks (MIN) constitute an effective trade-off in terms of cost and performance [11].

In MINs, communication takes place through a number of programmable switching elements (SEs). In a basic network, the set of the SE configurations allows to implement only a subset of all permutations of inputs into outputs. Moreover, there is only one path connecting a given input-output pair. Any fault making a given path not usable, actually reduces the number of permutations that can be applied. Redundant MINs overcome this limitation with limited overhead, thanks to their intrinsic path redundancy.

As shown in a previous work [6], the basic element, or *slice*, of our multi-sensor interconnection network is a 4x4 three-stage Clos network [12] made, of 2x2 SEs, which allows to connect four 1-bit inputs to four 1-bit output (see Figure 1). To interconnect 4 *n*-bit input/output pairs, *n* slices must be used. Each SE can be configured to apply either a *straight* or a *crossed* connection of its input/output terminals.

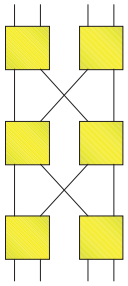


Figure 1. 4x4 three-stage Clos network with 2x2 SEs.

Differently from non-redundant networks (also known as 1-path networks), three-stage Clos networks allow to define multiple paths for each permutation. This feature plays a basic role in the definition of the fault-tolerance capabilities for the proposed system. Detailed properties about the Clos topology can be found in [12]. Three-stage Clos networks have two interesting properties: they are re-arrangeable and non-blocking. These two properties guarantee the possibility of applying all the permutations of inputs into outputs, that is, if k is the number of input and output nodes for the network, then the number of allowed permutations is $k!$.

Each slice has 24 possible permutations of its inputs into outputs, and for each of them there are at least two different network configurations (Figure 2).

Proceedings of the 2001 IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT'01)
1063-6722/01 $17.00 © 2001 **IEEE**

IEEE COMPUTER SOCIETY

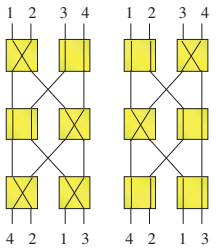


Figure 2. Example of permutation with 2 equivalent net configurations.

Furthermore those permutations, whose SEs in the intermediate stage have the same setting, have four equivalent network configurations (Figure 3).

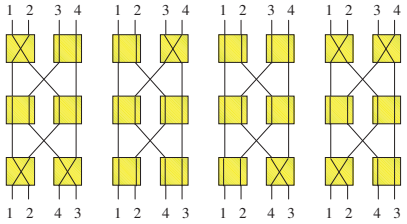


Figure 3. Example of permutation with 4 equivalent net configurations.

The net configuration activity is carried out by means of a proper control mechanism. At a high level of abstraction, the configuration process can be split into three phases. In the first, the assignment of the sink to source elements is accomplished. In the second phase, the set of permutations satisfying the requested assignment is identified. Finally, in the third phase, each slice is set with a configuration corresponding to the selected permutation.

3. System Implementation

Figure 4 shows the self-checking structure of a slice. Input and output bits for the slice are given to a *Parity Checker*(PC). Parity is actually the only invariant property of slice operation.

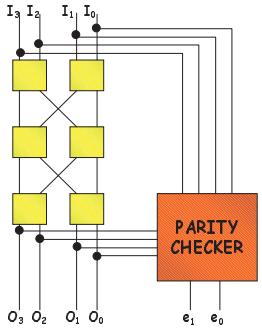


Figure 4. Slice self-checking structure.

The PC is endowed with self-checking ability so that it can reports faulty conditions relevant to the set of SEs, as well as faults within its own circuitry. The two output bits e_1,e_0 are encoded using two-rail code; in particular, the output pair (0,0) represents the fault free conditions, (1,1) denotes faults within the set of SEs , and (0,1) or (1,0) represent the cases of faults within the PC.

The MIN has been implemented by using a Xilinx 40xx family device [13]. Each SE requires a single CLB, thus six CLBs make up a slice. The PC takes three CLBs. The CLBs used to implement the slice and its PC lie on the same column of the FPGA's CLB array. Two slices can be placed in the same column, as shown in 5.

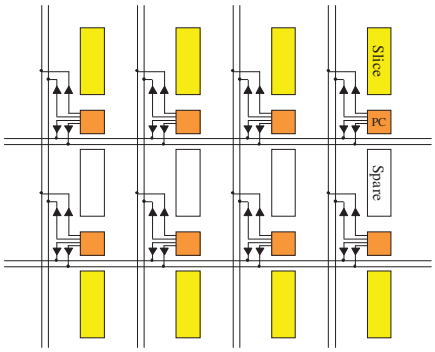


Figure 5. Slice and PC layout.

Outputs from each PC are connected to two pairs of open-drain buffers, one driving two vertical lines, and the other driving two horizontal lines. This allows to set a wired-OR connection between the two PCs on a column and a wired-OR connection among all the PCs over the same row. Anytime a PC reports a faulty condition (at least one of its outputs goes high, according to the chosen two-rail code), both the driven vertical and horizontal lines are set, thus making it possible to locate the faulty functional block (either a slice or a PC) within the implemented MIN. This diagnostic information allows to start the more appropriate repair procedure, according to the strategy described in the next section.

A group of six spare CLBs is placed within each column. It can be used to replace, in the event of a permanent fault, either a whole slice or the two PCs (one at a time). This spare block lies between the two working slice/PC clusters, in order to minimise the path delay after a possible reconfiguration.

4. Fault-Tolerance Strategy

The fault tolerance strategy adopted is based on a *on-line self-test and repair* approach and involves both network configuration and FPGA re-configuration. The former is applied whenever a fault affects a SE and allows a very quick fault repair, while the latter, requiring a longer time due to device re-configuration, is more suitable for permanent faults affecting the FPGA's global interconnection resources, the parity checkers, and any other fault that can be no longer repaired by configuring the MIN.

4.1 Network re-configuration

Re-configuration may involve either a single slice or the whole network, depending on the case. Specifically, when a fault affecting a given slice is detected, an attempt to fix the problem

is made by reconfiguring the faulty slice with an equivalent configuration. If no available configuration is found, or reconfiguring the slice fails due to the fault, a new trial permutation can be applied whenever allowed by the running application. The new permutation requires all the slices to be configured accordingly. Of course, the system may switch from a permutation to another not only in response to a fault, but also for application related needs.

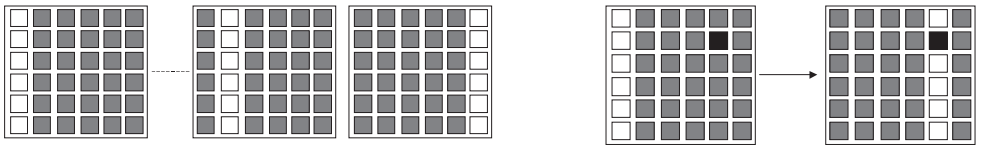
Slice re-configuration and network re-configuration are handled by two level control units: the *slice control unit* (one unit for each slice) and the *net control unit*. Each unit has its own fault table so that, anytime a given configuration/permutation gets faulty, a proper mark is put in the corresponding location of the relevant faults table, as to avoid using that configuration/permutation in a successive re-configuration step.

4.2 FPGA Re-configuration

As described in the previous sub-section, interconnection network re-configuration, either at slice level or at network level, is made by changing the type of input/output connection of SEs. Any fault not involving directly SE operation, or preventing SE operation permanently, needs to be treated in a different way. This is the case, for instance, of faults affecting the FPGA's configuration memory, parity checkers, links among SEs, and SE permanent faults, that require a re-configuration at the device level.

FPGA re-configuration is based on pre-compiled configuration programs, stored on-board, as proposed in [6]. The first step of the FPGA re-configuration strategy consists in the re-application of the original device configuration program. This allows to fix all the soft faults affecting the device configuration memory that could be induced by ionizing radiation. Anytime a fault is detected, the current MIN inputs (that is, the ones stimulating the fault) are latched. Such values are then given as input to the MIN after re-configuration to analyse its response. If the MIN still behaves incorrectly, then the fault is likely to be permanent.

Upon detection of a permanent fault, an alternative configuration is chosen among those available in the on-board configuration storage memory, depending on the kind of the fault. In case of permanent faults affecting SEs, or faults affecting the PCs, a configuration program which replaces the faulty block with spare CLBs available in the relevant column is chosen. If such spares are no longer available or when, for any reason, replacing the faulty block does not fix the problem, then the whole column must be replaced. To this end, a spare column is used in the original design mapping as shown in figure 6a, and a set of alternative program that move the spare column within the device is prepared. When the actual fault requires the column to be replaced, the relevant configuration program is applied, as shown in figure 6b. All the possible problems posed by re-routing the original design around a spare column have been addressed.



(a) **(b)**

Figure 6. (a) Set of alternative FPGA configurations; (b) use of the spare column to fix a fault.

5. Fault-Tolerance ability

The fault tolerance ability for a *k*-slice MIN has been evaluated by means of VHDL code simulation. All possible memory cell bit-flip have been considered, and specifically those affecting:

- LUTs used to implement the SEs;
- Configuration memory's cells within the CLBs used to map the SEs;
- Configuration memory's cells within the FPGA interconnection resources used to link SEs;
- Configuration memory's cells within the I/O blocks and relevant links to SEs and checkers.

The following general properties hold for the implemented MIN and have to be taken in care to completely analyse its behavior in case of faults:

- MIN slices have dedicated FPGA resources (LUTs, interconnects, I/OBs), so that a fault in a given slice has no effects on the others;
- Links among SEs of a slice make use of independent data paths, thus a faulty link does not affect the others;
- Any single bit-flip within a given slice (affecting either LUTs or interconnections) may result in a single bit output error for that slice.

According to the above general properties and the classes of considered faults, it has been verified that, for every injected fault, the system either behaves as if fault free or it reports the faulty condition; besides, there is always at least one input codeword that stimulates the fault. As far the parity checkers are concerned, they were designed as to have the *code disjoint* property, so that the whole system have the *totally self-checking* property.

The repair ability depends on the fault location within the FPGA. Particularly, faults affecting the SE's LUTs can be faced with either slice or MIN reconfiguration, while configuration memory's faults require a device level reconfiguration. Indeed, reconfiguring a slice means, in practice, changing input values for the LUTs used to implement the SEs (i.e., addressing different portions of the same memories), which has no effects on data paths (either within a CLB or in the interconnections among CLBs). Of course, not all the faults affecting the configuration memory cells result in system error, since only a portion of them is actually used to map a design into the device. It is quite difficult to evaluate the impact of unused memory cells on system errors as far as interconnection resources are concerned, since it may depend on the device size. Differently, the influence of unused memory cells within a CLBs can be evaluated. Figure 7 shows a simplified CLB structure with its configuration cells. Solid lines represent the data paths for mapping a SE into the CLB. As we can see, only those cells involved in such a data path can actually affect the SE operation (specifically, those configuring the two grey multiplexers). In practice, only 2 cells out of the 34 available (i.e., the 5.8 %) can cause, when faulty, a wrong SE behavior.

The repair ability analysis for faults affecting the LUTs used to implement SEs are summarized in Figure 8. According to the adopted fault model, each slice (thus the interconnection network) features a 100% fault-tolerance ability in case of single fault. For two successive faults, over 90% of all possible faults is tolerated by means of MIN re-configuration. For three, four and five successive faults, respectively the 75%, 55% and 35% of all possible faulty cases is tolerated. This percentage decreases down to the case of nine successive faults, for which less than 1% of the possible faulty cases is tolerated.

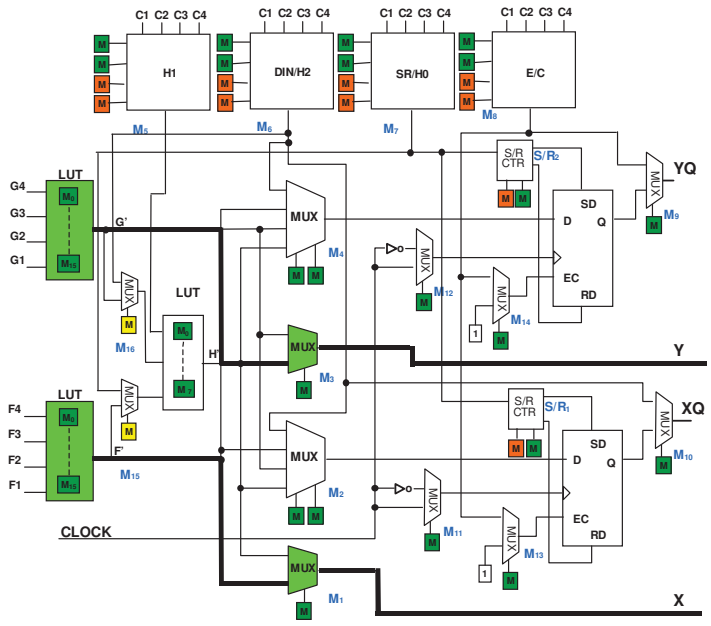


Figure 7. CLB structure of the XC4000 family.

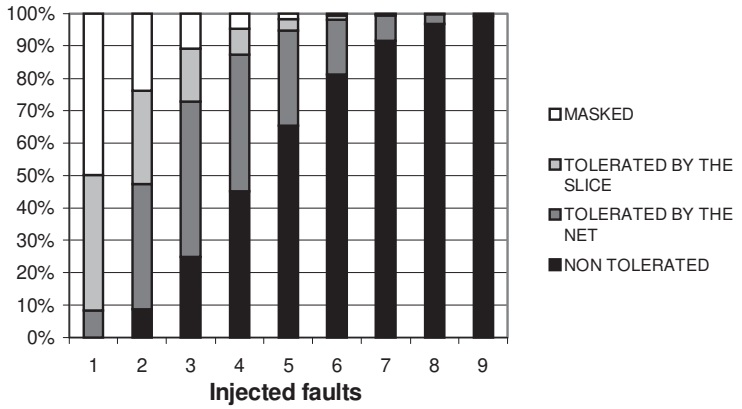


Figure 8. LUT fault-tolerance ability.

6. Conclusions

The paper presented an FPGA implementation of a multistage interconnection mechanism and illustrated the characteristics and its fault-tolerance ability. The fault-tolerance strategy is based on a on-line-test and repair approach. A twofold reconfiguration approach was proposed, which uses a net-level configuration and an FPGA-level configuration. The former allows to

Proceedings of the 2001 IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT'01)
1063-6722/01 $17.00 © 2001 **IEEE**

IEEE COMPUTER SOCIETY

recover the faulty condition in a very short time, while the latter allows to perform an accurate on-line diagnosis of the implemented MIN. The solution shown a high level of tolerance of faults affecting both CLBs and interconnections.

References

[1] M.A. Figuereido, P.H. Stakem, T.P. Flatley and T.H. Hines, "Extending NASA's Processing to Spacecraft", *IEEE Computer,* vol. 32, n.8, June 1999, pp.115-118.

[2] J.A. Rohr, "Software-Implemented Fault Tolerance for Supercomputing in Space", Fast Abstract at the *28th International Symposium on Fault Tolerant Computing Systems,* [Munich, June 23-25, 1998], IEEE Computer Society, http://www.chillarege.com/ftcs/fastabstract/373.html.

[3] J.C. Mather, "Proposal for Supercomputer Applications for NGST", *NGST Technical Report*, available at http://www.ngst.nasa.gov/public/unconfigured/doc_0101/rev_02/proposal.html.

[4] M.A. Nieto-Santisteban et al., "On-board Supercomputing for NGST and NASA's Remote Exploration and Experimentation Project", *NGST Technical Report,* available at http://www.ngst.nasa.gov/public/unconfigured/doc_0559/rev_02/NGST_REE.ps.

[5] E. Bezerra, F. Vargas and M.P. Gough, "Merging BIST and Configurable Computing Technology to Improve Availability in Space Applications", *Proc. 1st IEEE Latin-American Test Workshop*, (LTW'00) [Rio de Janeiro, Brazil, March 13-15, 2000], also available at http://www.sussex.ac.uk/Users/tapu9/publication/ltw00.pdf.

[6] M. Alderighi, F. Casini, S. D'Angelo, D. Salvi and G. R. Sechi, "A Fault-Tolerance Scheme for a MIN-based Multi-Sensor System", *Proc. IEEE Symposium on Field Programmable Custom Computing Machines*, [Rohnert Park, California, April 30-May 3, 2001], IEEE CS Press, in press.

[7] K. Roy and S. Nag, "On Routability for FPGAs under Faulty Conditions", *IEEE Trans. on Computers*, vol. 44, n. 11, November 1995, pp.1296-1305.

[8] F. Hanchek and S. Dutt, "Methodologies for Tolerating Cell and Interconnect Faults in FPGAs", *IEEE Trans. on Computers*, vol. 47, n. 1, January 1998, pp. 15-33.

[9] J. Lach, W.H. Mangione-Smith and M. Potkonjak, "Low Overhead Fault-Tolerant FPGA Systems", *IEEE Trans. on VLSI*, vol. 6, n. 2, June 1998, pp. 212-221.

[10] J. Emmert, C. Stroud, B. Skaggs and M. Abramovici, "Dynamic Fault-Tolerance in FPGAs via Partial Reconfiguration", *Proc. IEEE Symposium on Field Programmable Custom Computing Machines*, [Napa Valley, California, April 17-19, 2000], IEEE CS Press, pp. 165-174.

[11] G.B. Adams III, D.P. Agrawal, H.J. Siegel, "Fault-Tolerant Multistage Interconnection Networks", *IEEE Computer,* June 1987, pp. 14-27.

[12] V.E. Benes, "*Mathematical theory of connecting networks and telephone traffic*", Academic Press, London, 1965.

[13] Xilinx Inc., *"The Programmable Logic Databook",* 1998.