Preprints of the 30th IFAC Workshop on Real-Time Programming and
4th International Workshop on Real-Time Software, pp. 131 – 136

Reconfigurable Network on Chip Architecture for Aerospace Applications

Kasra Motamedi*, Nicholas Ioannides*, Mark H. Rümmeli, Igor Schagaev***

** Faculty of Computing, London Metropolitan University, 166-220 Holloway Road, London, N7 8DB, UK*
Email: {KAM0626, n.ioannides, i.schagaev}@londonmet.ac.uk
*** IFW Dresden, P.O. Box 270116, D-01171 Dresden, Germany*
Email: m.ruemmeli@ifw-dresden.de

Abstract: A fault tolerant network on chip (FT-NoC) system with reconfigurable architecture for aerospace applications is proposed. Applying different types of redundancy on chip increases reliability, efficiency and effectiveness of the NoC and, at large, the aircraft control system itself. The central theme is the application of redundancy to tolerate hardware faults in the processor cores of the NoC system's functional units. Second theme is the implementation of an application-specific configuration topology for the aircraft control system. The elements inside the chip are designed by considering reliability, fault tolerance and power consumption. Time, information and structural redundancies are implemented to achieve tolerance against potential permanent or transient faults. The proposed NoC system benefits aircraft control systems through higher reliability, parallelism in wire speed, reconfigurability of hardware for real time data processing, and also by lowering power consumption.

Keywords : Reliability, Fault Tolerance, Redundancy, Network on Chip (NoC), Reconfigurability, Application Specific NoC, Avionics, Aircraft Control Systems, ONBASS project

1. INTRODUCTION

The International Technology Roadmap for Semiconductors (ITRS) predicts that before the end of this decade single Systems on Chip (SoC) could embed 4 billion transistors using 50nm technology, operating at 10GHz each (Benini and De Micheli, 2002) . These advancements raise problems in the communication and interconnection infrastructure between the components inside the chip, hence new architectures and scalable design approaches are needed. In order to cope with the growing needs of the interconnected infrastructure the Network on Chip (NoC) concept has been introduced which benefits computer systems by providing higher levels of performance and reliability. Such computers are now a mandatory component in the design of automatic control units for aviation systems.

In this paper we propose a Fault Tolerant NoC (FT-NoC) system specifically designed for the control of aircraft parts by implementing redundancy in the topology of the NoC thus increasing reliability. In doing so, we review the architecture of the elements inside a NoC system; classify various faults and redundancy types to implement fault tolerance; and briefly mention various parts of an aircraft attempting to map the structure of an aircraft within the design of a FT-NoC system for aircraft control. Following, the proposed FT-NoC system, application specific and embedded reconfigurable architecture, choice of processing core, and future work are discussed.

2. NoC SYSTEM

The Network on Chip system is a collection of computational resources connected together through a network inside the chip and communicate using packets. The communication architecture consists of interconnected switches each connected to a resource which can be a processor core, a memory block, or even a custom designed hardware which is generally called Intellectual Property (IP) Block (Ning, et al. 2007). For avionics and aircraft control systems, IP blocks can be sensors, analogue to digital converters (ADC), etc. One of the main advantages of NoC systems is the separation of computation and communication in these systems. The communication units are Network Interfaces (NI) and switches. NI act as the middle layer and transform streams of bits from the computational resources into packets before sending them to a router or switch and vice versa.

2.1 Switching

Switches are used to route packets over the network using different techniques. Recent packet-switching trends show that wormhole switching is the best choice for NoCs (Pande et al. 2005). Switch design also depends on the routing scheme chosen.

Routing can be deterministic, adaptive or hybrid. Deterministic routing algorithms always provide the same path between a given source and destination whereas adaptive routing algorithms use information on routing traffic or channel status to

avoid congested or faulty parts of a network when determining the path. The better choice is the Hybrid Dynamic (HD) routing protocol which uses predefined paths (deterministic routing), gets updates from its neighbours (distance vector), and allows each switching router to calculate best paths based on their understanding of the network (link state). A dynamic routing protocol in a NoC is necessary to enable an acceptable level of fault tolerance in a reliable chip. Furthermore, switches should be able to implement these techniques without consuming too much area and power.

2.2 Topology

Network topology defines the placement and interconnection of nodes inside the NoC area and determines the bandwidth and latency of a network (Salminen et al. 2008). The most common topologies are identified as the 2D Mesh and Torus due to their grid-type shapes and regular structure (Hu et al. 2008). These are the most appropriate topologies and formations for a two dimensional layout on a chip when an application specific topology is not considered.

These selected topologies are based on the routing hop count, redundancy overhead in number of links in case of link failure, link lengths, energy consumption over the links and switches, and finally area usage over the silicon surface. The Torus topology introduces long wires (link redundancy) among the last nodes to complete the shape of the topology. Employing long wires in very large scale integration (VLSI) and deep submicron domain (DSM) systems increases the capacitance among wires, influences the inductance of links, and results in development of crosstalk over links. The other promising topology formation for FT-NoC is the application specific architecture – a subject of this paper.

3. FAULT TOLERANCE AND REDUNDANCY

Fault tolerance is a particular technique that enables the building of systems that maintain the expected service despite the presence of errors caused by hardware faults within the system itself. The use of redundancy increases the reliability of the system but also affects its performance and increases the application costs (power usage and area consumption). A balanced trade off among these factors must therefore be considered for maximum performance and high level of fault tolerance.

3.1 Fault Classification

Faults are classified in three major groups: design faults, manufacturing faults, and operational faults (Weaver and Austin, 2001). Operational faults, based on their frequency and probability of occurrence, are divided into permanent, intermittent, and transient (Ali et al. 2007). They may also be caused by different environmental, operational, and technological processes (De Micheli and Benini, 2006) . A major concern in a fault tolerant NoC design is the tolerance and redundancy of the system against permanent and transient faults caused during operation. Transient faults or malfunctions occur regularly and can be tolerated even at the instruction level (Schagaev, 2008). An example is when some area of the chip experiences an internal failure with permanent effect. However, both types of fault cannot be easily correlated to any specific operational, environmental or technological condition.

3.2 Redundancy Classification

Redundancy in computer systems can be classified in terms of *time*, *information* and *structure* (Schagaev and Zalewski, 2001). Any of these redundancy types can be applied to system hardware or system software to protect the system against various types of faults and to increase the reliability of the system. Information redundancy can be realized by introducing coding techniques for parity check into the data stream and packets. Implementation of redundant hardware for simultaneous execution of same data on various channels and comparing the outcomes is a frequent type of structural redundancy.

3.3 Fault Tolerance in NoC using Redundancy

There are several potential forms of fault tolerance implementations in NoC systems. Segmentation of the communication and computational infrastructure of NoC systems, one of its core concepts, provides inherent solutions to the reliability problems among different components and areas of systems. For information redundancy, information may be prioritized based on the attention needed by the network infrastructure for the safety and integrity of data into three classes: latency critical, data streams and miscellaneous information (Bjerregaard and Mahadevan, 2006). Each group has its own type of coding technique for parity check.

Most common faults in the structure of the system are noise concerns, technology delays and fabrication faults in the manufacture of NoC integrated circuits (IC) (De Micheli and Benini, 2006) . The self-calibrating method was a solution to tolerating the gate delay (Worm et al. 2005). For noise concerns, packet encoding and redundant transmission of information has been introduced. Inserting extra links and wires would tolerate the manufacturing faults but would compromise the performance and energy consumption considerations inside a NoC IC.

4. AIRCRAFT PARTS

Each aircraft, commercial or military, consists of structurally different parts. Checking the status and operating conditions of all these parts is critical for flight safety. These parts, shown in Figure 1, are:

- Control system: A collection of electronic and mechanical equipment which control aircraft with accuracy and consists of cockpit controls, sensors, actuators and computers.
- Cockpit: The Captain Cockpit is the major central part for controlling and navigating any aircraft. Information from each section is gathered here for monitoring and

piloting the aircraft by automatic systems or human pilot.

- Black Box: These are recorders which store records of status and condition of every part and component during a flight.
- Wings: Main parts of an aircraft for applying the lifting forces and fluttering the aircraft. Other main parts integrated into wings are flaps and spoilers for controlling the cruising speed.
- Engines: Provide the thrust force to push the airplane forward through the air.
- Gears and fuel tanks: Takeoff and landing are the most critical aspects of any flight; during mentioned flight modes gear status must always be monitored.
- Tail, elevator and rudder: These parts provide the ability to change forces for a means of controlling and maneuvering.
- Fuselage: The main body of any aircraft which contains the arsenal or passengers and cargo. Conditions inside the fuselage must be checked continuously (GRC, 2008).

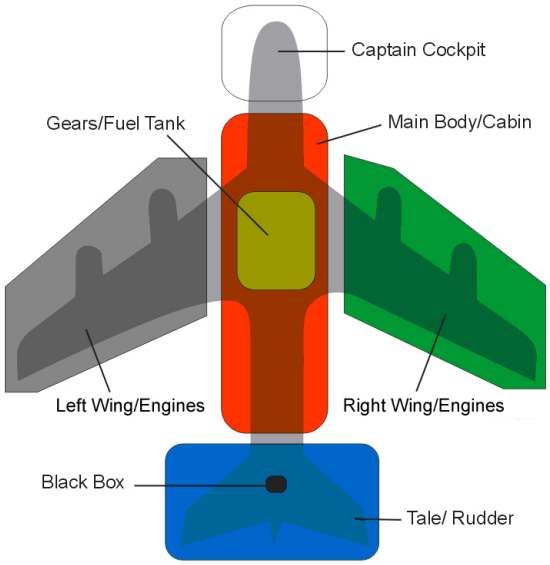

Fig. 1: Aircraft parts

All these parts must be controlled by aircraft controlling and piloting systems which rely on mechanical and hydraulic instruments positioned between the aircraft's maneuvering parts and the flight surfaces o n the fuselage, wings and tail. On the other hand, computer based control systems enable novel aircrafts to save weight and improve reliability. These are called Fly-By-Wire (FBW) systems and achieve their higher levels of reliability by replicating sensors, computers and actuators. Redundancies in hardware and software provide graceful degradation in the event of system failure or fault. In a degraded state, essential functionalities remain available allowing the pilot to continue the flight and land the aircraft safely. A typical FBW implementation uses seven separate computers with different types of software to provide redundancy. Some computers are used as primary and others as backup in case of any failure or fault in the primary systems. But the installation of such high number of hardware consumes space and electricity and increases the weight of the aircraft without improving on the reliability as it is still inefficient in tolerating permanent faults in hardware and power supply.

5. FT-NoC FOR AIRCRAFT CONTROL SYSTEMS

The first phase in designing an aircraft control system using NoC is to choose the objectives and define any constrains. Each part must be controlled and monitored by its own subsystems and must be mapped onto one functional unit in the FT-NoC system. All sections must be equipped with their own processing unit and memory in order to maximize the performance and reliability of the whole control system. To address these requirements this paper discusses:

- Power consumption in components and links;
- High levels of reliability by implementing fau lt tolerance through redundancy at various levels especially in the different processing cores of the NoC;
- Embedding reconfigurability in the network topology structure.

5.1 FT-NoC with Fixed Topology

Based on the objectives, each major aircraft element is mapped onto a functional unit in the NoC. Each unit consists of the processing unit (Embedded Reliable Reduced Instruction Processor - ERRIC), the local memory and any other necessary IP block. Such a topology, a fixed 2D-mesh topology FT-NoC, is shown in Figure 2.

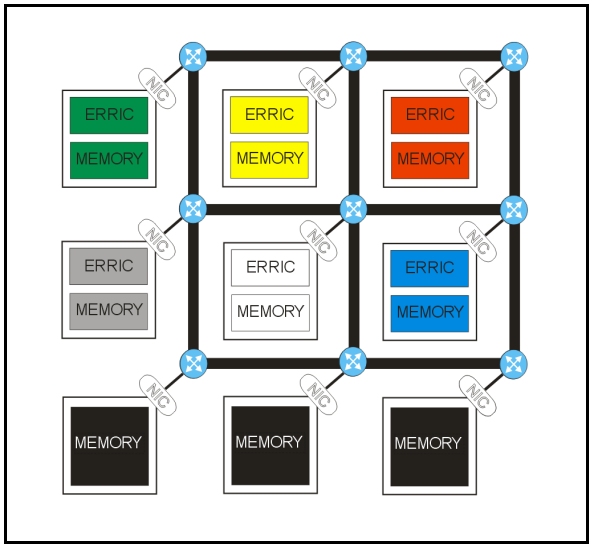

Fig. 2: FT-NoC with fixed 2D-Mesh topology

Figure 2 shows that each major aircraft structural element presented in Figure 1 has its own colour coded functional unit. Parts with similar traffic pattern and functions are placed as close as possible to each other to reduce the network overhead and decrease the hop count during packet transmission. For state condition recording purposes and to keep the formation resemblance, two extra memory blocks are used. One of the most important advantages in designing NoC for multiprocessor systems is that the information

which is inserted in the system from any input element (sensors, etc) or headed to an output element (altimeter, etc) are directly connected to their respective processing units. The main disadvantage of the 2D-Mesh system is the delay on the system incurred by the routing paths. And even though the cockpit, the most important unit with centric functions, is placed in the centre of the matrix topology, some hop-count delays from the centred unit are still present.

5.2 Application Specific Architecture for FT-NoC

Having a centric unit when architecting a system entails the implementation of a star network topology with the central unit, in this case the cockpit switch, being directly connected to all units thus decreasing the hop count in routing paths and increasing the bandwidth available over each directly connected link. However, the star topology's main disadvantage is its low fault tolerance due to a link failure or a switch crash resulting in whole network crashes and all communication between the various components stopping. Under these conditions the system software will have the ability to reconfigure the topology when a fault in the links is detected. The system can modify the architecture and switch it to ring in four cases:

- If a fault is detected over any of the star links the redundant ring links will come out of stand by and form a shortest path to the central switch.
- If a link is jammed due to traffic congestion and shortage of bandwidth in a link the central switch will attempt to form another path to the destination and start load balancing over the directly connected and the redundant links.
- To decrease the switching load in the central processor when the source and destination of transmissions are adjacent, the central switch will bring up the direct link amongst them thus avoiding the communication through it (useful for writing information into the Black Box).
- If the central switch malfunctions the topology will change completely to ring formation and the idlest processing unit will take the control of the network. The system states will then return to the latest healthy status of the network stored in the Black Box memory.

The system architecture for a FT-NoC is shown in Figure 3.

In this case e ach functional unit relates to a part of the aircraft and is integrated with a direct connection from its switch to the processing units, the ERRIC, and another direct connection to the memory elements inside that unit. These connections guarantee that for as long as a memory or a processor is still available in the system, the whole system can continue to function.

The major technical aspects and design consideration of this system are:

- The functional units are composed of a processing unit (ERRIC processor structure), a local memory module and their corresponding Intellectual Property (IP) blocks. IP blocks import and export data from and to the peripheral instruments or ADC/ DAC sensors.
- Structural redundancy is realized in the black box memory architecture through two sets of redundant links connecting it to the central switch.
- The six links form a star topology as the main active formation in the system. Redundant links are placed in the system to enable embedded architecture reconfigurability and load balancing in the network by forming a ring topology inside the NoC system chip.
- The system software implements a generalised algorithm of fault tolerance, known as GAFT, to detect permanent links or switches and reconfigure the network if necessary.
- The functional units with similar tasks and traffic patterns must be placed as close to each other as possible to enable direct communication between them.
- Even though the use of straight links reduces the delay on the network by keeping the lengths of these links short implementing such straight links must be avoided so as to reduce crosstalk on the links connecting the central unit to each functional unit.
- Real time system Recovery Point (RP) could be stored as a part of the Black Box.

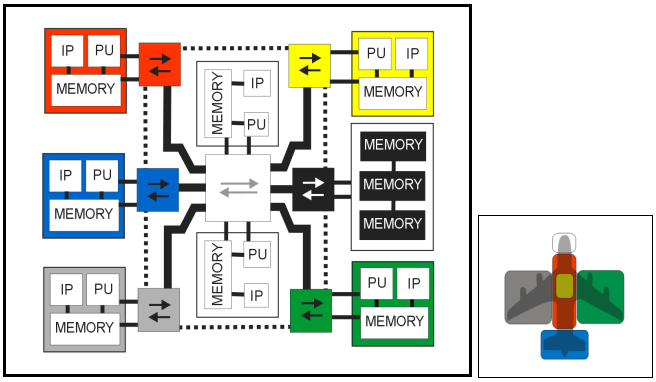

Fig. 3: FT-NoC system with ERA

Such specific computer system design and architecture for aircraft control systems will provide:

- Low levels of power consumption in the system,
- Parallelism in hardware and wire speed by increasing the available resources,
- Reliable system for sensitive computation through implementation of fault tolerance in various levels,
- Low levels of system maintenance by providing auto-reconfigurability,
- Compatibility with previous system architectures and designs installed in current Fly-By-Wire (FBW) systems.

6. FAULT TOLERANT PROCESSOR FOR NOC

Reliability apprehension in data processing inside the PUs grows since the use of Deep Sub-Micron (DSM) fabrication technologies increases architecture complexity, amplifies exposure to natural radiation sources and adds noise-related

fault tolerance and redundancy mechanisms (Weaver and Austin, 2001).

6.1 The ERRIC Processor

To counter the reliability challenges inside the processing units of a NoC system we use the Embedded Reliable Reduced Instruction Computer (ERRIC) (ONBASS Project, 2007). The instruction set for this processor is specially designed malfunction tolerant and fail-safe for permanent faults. This maximizes performance and reliability of the NoC since the reliability of a system is based on the reliability of its components (Schagaev, 2008). Ability to tolerate malfunctions invisible for the rest of the system is crucial as the ratio of malfunction/permanent faults in aerospace is of the order of 10^{5} (ONBASS Project, 2007).

The ERRIC basic elements for instruction manipulation are shown in Figure 4. They include a Control Unit (CU) to decode an instruction and fetch the next one, Register Files (RF) to keep data, and separate Arithmetic Unit (AU) and Logic Unit (LU) to execute arithmetic and logic functions.

The extra hardware blocks check inputs and recover from any detected fault (marked with a bright colour in Figure 4). Structural redundancy used to achieve malfunction tolerance is about 13%. The error recovery is only activated when a fault has been detected so the power consumption stays as minimum (Schagaev, 2008).

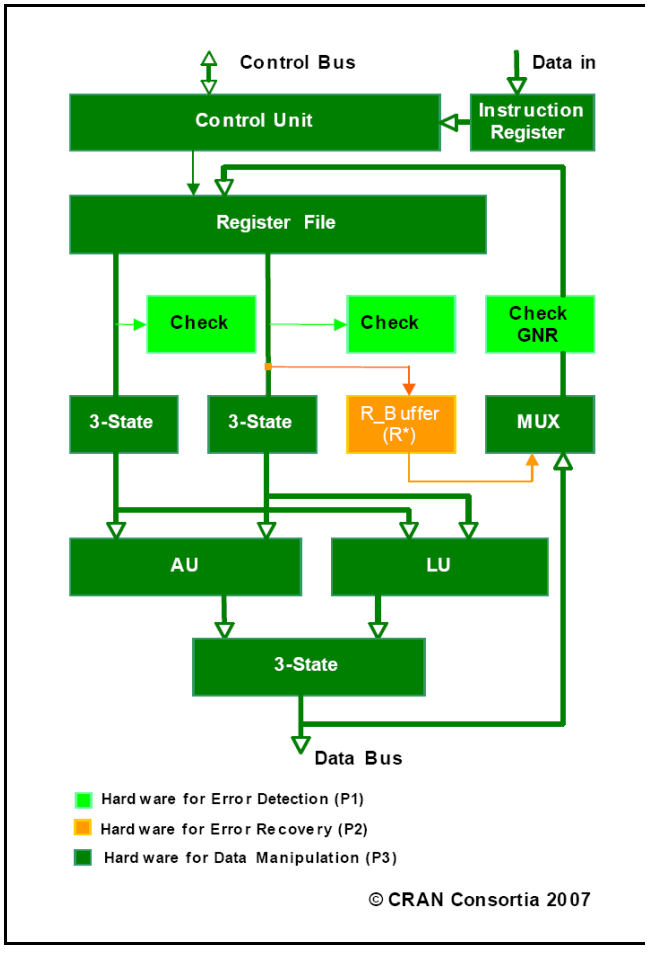

Fig. 4: The ERRIC structure

The ERRIC is divided to passive and active zones enabling the application of different redundancy techniques. The operands are checked during operation and if not damaged the operation is performed with the result either stored back into the RF with extra information inserted for future checking, or, if output, it is transmitted to memories or external I/O devices. The ERRIC is a 32 bit architecture, and is able to fetch 2 instructions in every fetch cycle which reduces the execution time to nearly half that of other processors.

6.2 Performance Comparisons

The ERRIC performs better when compared with other processors. The necessary number of gates and registers needed by ERRIC has been evaluated by using Altera Quartus II software on a Cyclone II FPGA. Here we assumed that each single logical element is made up of 6 to 10 transistors on average. Table 1 shows the total number of elements used by the ERRIC with and without the recovery mechanism and fault tolerance.

Table 1: Hardware consumption by ERRIC with and without redundancy

	Total Logic Elements	Total Registers	Total Memory bits
ERRIC without fault tolerance features	521	158	2,048
ERRIC with fault tolerance features	586	173	2,112
Redundant Physical Overhead	12.48%	9.50%	3.13%

The final expected ASIC Implementation of the ERRIC processor will also consume less than 10,000 transistors. Implementations show that the redundant hardware needed for the realization of fault tolerance is less than 13%. Also, the total number of memory bits essential for the execution of tolerance against faults through redundancy is only 3%. Furthermore, the number of transistors used to realize the ERRIC processor compared with other processors is shown in Table 2.

Table 2: Number of transistors in Intel and ARM processors

Family	Trade Name	Date Introduced	Number of Transistors
80786	Itanium	May, 2001	300 million
ARM	ARM7TDMI	2006	100,000

The numbers shown in Tables 1 and 2 clearly indicate a reduction in the amount of gates required to implement the following processors:

- ERRIC / Intel Itanium = 1/30000
- ERRIC / ARM7TDMI = 1/10

It is obvious from the above that the use of ERRIC in FT-NoC systems reduces the number of gates in the die area and the power consumption. It also increases the number of active processing cores of NoC.

The FPGA based ERRIC prototype shown in Figure 5 gave positive and promising results.

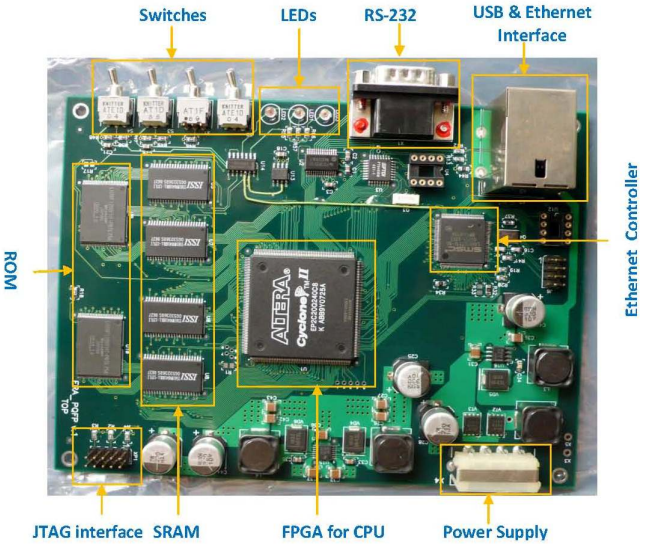

Fig. 5: The ERRIC prototype on FPGA hardware

7. CONCLUSIONS AND FUTURE WORK

A Fault Tolerant NoC system, specifically designed for the control of aircraft parts by implementing redundancy in the topology of the NoC has been proposed.

NoC based systems separate computation and communication between the various elements thus providing inherent solutions to the reliability problems among different components and areas of systems.

Ap plication specific design of FT-NoC involves mapping onto a functional unit in the NoC and must be integrated with two direct connections from its switch, one to the processing units and another to the memory elements inside that unit.

The processing units are based on ERRIC (chosen for its superior performance) and the network topology is controlled by an embedded reconfigurable architecture, which is both star (primary) and ring (backup). These connections guarantee that for as long as at least one memory and one processor are still available in the NoC, the whole system can continue to function.

A prototype of the system built on FPGA hardware gave positive and promising results in terms of performance and reliability. Further work will focus on monitoring the behaviour of the prototype under various conditions and consideration will be given to which may be the best hardware platform (CPLD, ASIC or Wafer) for this type of FT-NoC system.

REFERENCES

Ali, M., Welzl, M., Hessler, S. and Hellebrand, S. (2007) A Fault tolerant mechanism for handling Permanent and Transient Failures in a Network on Chip, *Int'l Conf. on Information Technology (ITNG)* .

Benini, L and De Micheli, G. (2002) Networks on Chip: A New SoC Paradigm*"*, *IEEE Computer*, vol. 35, no. 1.

Bjerregaard, T. and Mahadevan, S. (2006) A Survey of Research and Practices of Network-on-Chip, *ACM Computing Surveys*, Vol. 38; 2006

CRAN Consortia (2007) Control Reconfigurable Aircraft Network (CRAN) – Small or medium-scale focused research project (STREP) proposal (www.it-acs-ltd.com)

De Micheli, G. and Benini, L. (2006) Networks on Chips: Technology and Tools, *Morgan Kaufmann.*

GRC - NASA, (2008) Airplane Parts Definitions and functions, Glenn Research Center, *http://www.grc.nasa.gov/WWW/K-12/airplane/airplane.html*

Hu, W.H., Lee, S.E. and Bagherzadeh, N. (2008) D-Mesh: A Diagonally-Linked Mesh Network-on-Chip Architecture, *NoCArc, 1 st Int'l Workshop on Network on Chip Architectures held in conjunction with MICRO-41* .

Ning, W., Fen, G. and Qi, W. (2007) Simulation and Performance Analysis of Network on Chip Architectures Using OPNET, 7th Int'l Conf. on ASIC, pp 1285-1288.

ONBASS Project (2007) ON-Board Active Safety System Project, Active system safety implementation: hardware design, development and analysis.

Pande, P. P., Saleh, R., Grecu, C., Jones, M. and Ivanov, A. (2005) Performance Evaluation and Design Trade-Offs for Network-on-Chip Interconnect Architectures, *IEEE Transactions on Computers*, Vol. 54, Issue 8, pp 1025-1040.

Salminen, E., Kulmala, A. and Timo D. Hämäläinen, T. D. (2008) Survey of Network-on-Chip Proposals, *White Paper, OCP-IP* .

Schagaev, I. and Zalewski, J. (2001) Redundancy Classification for Fault Tolerant Computer Design", *IEEE Int'l Conf. on Systems, Man and Cybernetics*, pp 3193–3198.

Schagaev, I. (2008) Reliability of Malfunction Tolerance, *Proceedings of the IMSCIT*, pp 733-737.

Weaver, C. and Austin, T. (2001) A Fault Tolerant Approach to Microprocessor Design, *The Int'l Conf. on Dependable Systems and Networks (DSN'01)*, pp 411-420.

Worm, F., Thiran, P., De Micheli, G. and Ienne , P. (2005) Self-calibrating networks-on-chip , *Int'l Symposium on Circuits and Systems (ISCAS).*