in Proc. IEEE Conf. on Computer Design: VLSI in Computers & Processors, Austin, Texas, October 5-7, 1998, pp. 384-389

Fault-Tolerant Architecture for High Performance Embedded System Applications

Gul N. Khan
Division of Computing Systems, School of Applied Science
Nanyang Technological University, Nanyang Avenue, Singapore 639798
Email: asnkgul@ntu.edu.sg

Abstract

The architecture of a fault-tolerant embedded computer system is presented. It employs multiple processors for high performance and dual-port memory units for interprocessor communication. The high performance embedded computer (HPEC) system consists of five processors that are partitioned into two sets namely the computing and IO partitions. The computing partition is concerned with computational intensive tasks and it consists of three worker processors. The IO partition performs general-purpose and real-time I/O related tasks. It has two interface processors with high-speed I/O and fast interrupt capabilities. The processor cores for these partitions are selected according to computational and high-speed I/O functions. The HPEC system size can be adjusted for varying needs of computing and real-time I/O without affecting the basic architecture features. The HPEC architecture is fault-tolerant in terms of fault containment and isolation of faulty units. Reliability modeling and analysis of the system indicates that it degrades gracefully under different fault scenarios.

Key words: *safety-critical embedded systems, hardware and software fault-tolerance, high-performance embedded computers, parallel computing.*

1. Introduction

The embedded computer systems are being employed in simple consumer products like microwave oven, washing machine, and cellular phones as well as in computationally intensive products like laser printers and videophones. At the high performance end of embedded systems, safety-critical applications in the areas of avionics, astronautics and robotics demand fault-tolerance as well as high performance. So long the typical embedded systems have been small and execute only a few thousand bytes of code. Modern embedded computer systems may include megabytes of code and run at ultra speed to meet tight performance and reliability deadlines. The architects of embedded system are facing high throughput and reliability demands that have never before been required of these systems [1]. One can add a high performance CPU to handle a number of tasks but there are pitfalls in using powerful CPUs in real-time environment. Fast processors tend to have caches and memory managers that can easily increase the already long interrupt latencies. Increasingly, the system designers are responding with multiple processor solutions [2].

Future safety critical-control embedded systems are likely to have fault-tolerance and high-throughput requirements that current single processor embedded systems cannot meet. Typically, the allowable system failure probabilities are moving upward from 10^{-5} per hour to 10^{-10} per hour. Another main requirement is the high level of computing performance that not only includes high throughput and large memories but also the adaptability of the architecture to varying requirements of real-time critical applications. The target architecture should be able to adapt itself to the varying needs by trading performance with reliability and vice versa. A similar high performance embedded computer (HPEC) system architecture has been investigated and designed which employs dual-port memories for high speed interprocessor communication.

2. HPEC Architecture

The architecture of HPEC system, depicted in Figure 1, is aimed at high performance as well as safety-critical embedded applications. The system consists of five processors, which are fully connected using ten dual port memories **DP_{ij}** for i, j = 1, 2, 3 ---- 5 (i < j). In an **n** processor system, **n(n-1)/2** dual-port memories are needed for full processor connectivity and each processor has access to **(n-1) DP** memories. HPEC has three worker processors, **WP** dedicated to computing intensive tasks and two interface processors, **IP** that are responsible for general purpose and real-time I/O. The interface processors also perform system monitoring, fault containment and system recovery from failures.

The high performance of a multiprocessor system depends not only on using faster and more reliable hardware but also on efficient interprocessor communication. Dual port memory based processor interconnection provides a high-speed communication media. Dual-port memories have been previously used for interprocessor communication in large-scale parallel systems [3]. In HPEC, each processor bus is dedicated to one processor only and this has avoided the time-shared bus bottleneck by eliminating memory access conflicts. The HPEC architecture provides efficient routing schemes including single step broadcast supported by writing the message to multiple **DP** memories concurrently [3].

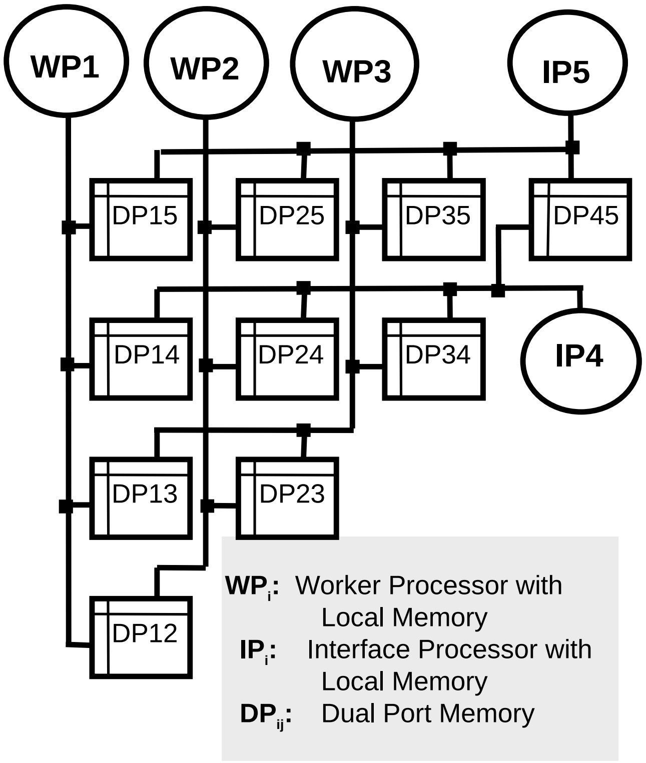
WP1
WP2
WP3
IP5
DP15
DP25
DP35
DP45
DP14
DP24
DP34
IP4
DP13
DP23
DP12
WP$_i$: Worker Processor with Local Memory
IP$_i$: Interface Processor with Local Memory
DP$_{ij}$: Dual Port Memory

Figure 1: HPEC System Architecture

3. High Performance, Fault-Tolerant Features

The prototype version of the HPEC system consists of five processors and it is partitioned into two sets of dedicated processors for achieving fault-tolerance as well as high performance. The computing partition performs high performance computation and it consists of three worker processors as shown in Figure 2. The second partition performs real-time I/O and it has two interface processors to communicate with the outside world. The processor cores for each partition can be selected according to their functions and may also be optimized for their respective tasks. Additional processors can be added to one or both of the partitions for varying computing and fault-tolerance demands without affecting the basic features of HPEC.

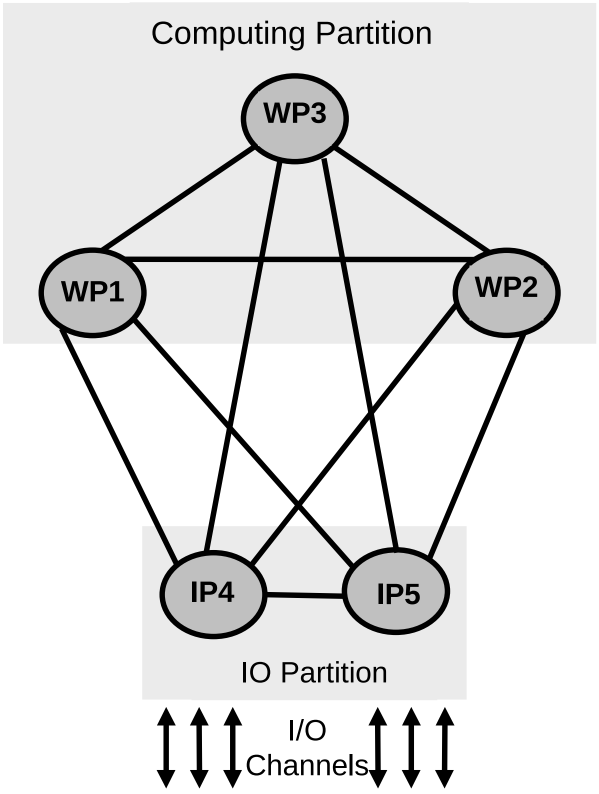

Figure 2: Logical Architecture of HPEC

3.1. Fault Detection

Fault-tolerant mechanisms employed in HPEC are depicted in Figure 3. Different techniques are used to detect faults in different units of the system. A watchdog timer each detects processor failure and generates processor fail (PF) signal. The local memories are equipped with error detection and correction circuits and in the case of a failure, they generate memory fail (MF) signal. In this way, each processing node is self-checking and declares itself faulty when the processor or its program memory fails [4, 5]. The **DP** memory and its interface failure are detected at the time of data transfer by using CRC/checksum errors.

The interface processors **IP_4** and **IP_5** provide fault tolerant support in addition to performing real-time I/O, load balancing and scheduling tasks. At a given time, one of the interface processor is designated as system controller to monitor and isolate faulty processors, **DP** and other memories. The second interface processor monitors the designated controller and takes over the charge of system controller in case the designated controller fails.

3.2. Fault Containment and Recovery

Fault containment and system recovery techniques are explained using Figure 3. The failing processing node interrupts the system controller (**IP_4** or **IP_5**) that performs necessary actions for fault isolation and system recovery. The failure of a processor or its local memory is handled as a single fault and system controller performs the following actions:

- It disables the interrupting capability of the faulty processing node.
- It generates the isolation signal, **IS_n** to inhibit the faulty processors from modifying the shared data or program in **DP** memories.
- The system controller broadcasts the failure of a processing node to rest of the system.
- It invokes a diagnostic process that can reset and analyze the failed processor. For transient faults, the failing processor is put back into service. Otherwise, the faulty processor is permanently kept out of service and its tasks are re-distributed to other healthy processors. Both interface processors keep a record of the useful work performed by other processors and in case of a processing node failure, its tasks are rolled back to a predetermined state.
- For a faulty local memory, the system controller also isolates its processor before diagnosing the memory. The diagnostic programs are executed from the corresponding **DP** memories. The processor with a faulty local memory is utilized in a degraded mode by executing the critical tasks that can fit into its **DP** memory blocks.

In the case of a **DP** memory failure, the system controller isolates it from rest of the system. The processor interconnection network facilitates alternate routes for interprocessor communication when a particular **DP** memory unit or its interface fails. For instance, if **DP_{ij}** memory fails, the communication between **P_i** and **P_j** processors is established through a third processor **P_k** by using **DP_{ik}** and **DP_{jk}** memory units. A high degree of dynamic redundancy exists for interprocessor communication.

The HPEC system architecture suits to most of the software fault-tolerant strategies including recovery block, N self-checking and N-version programming [6]. The computing partition of the system can be considered as a TMR system where each **WP** processor executes different versions of the application code and one of the **IP** processor works as a voter. Similarly in the IO partition both interface processors can be employed to work in duplex self-checking configuration. Reliable operation of the IO partition is essential and therefore, distributed recovery block scheme [7] should be employed for interface processors. Distributed recovery block scheme handles both software and hardware faults in a uniform manner. The HPEC system provides basic computing hardware units that can be configured and programmed to implement various fault-tolerant strategies for varying reliability and safety requirements.

4. HPEC System Reliability

There are two extreme cases for defining reliability of multiple processor systems. The parallel systems with large number of nodes (processor, memory and interface) have hundreds of switches, pins and wires of interconnection network and accurate communication requirements combined with thousands of lines of system level code. At one extreme, the probability that such a system is completely operational is very low. On the other extreme, one may claim that as long as two system nodes are working and communicating successfully, the parallel system is operational. However, a realistic reliability model of a parallel system like HPEC would require only two (one in each partition) fault-free communicating nodes for the system to be considered operational.

Threshold reliability for shared memory parallel systems has been introduced and analyzed by Hwang and Chang [8]. For a *P* processors and *M* shared memory units system, threshold reliability $R_{p,m}(t)$ is defined as the probability of having at least *p-out of-P* processors communicating with *m-out of-M* memory units in a time interval *(0, t)*. Threshold reliability concept is useful for evaluating degradable computer systems.

4.1. Evaluation of Graceful Degradation

The dual-port memory organization of HPEC can be considered as a restricted shared memory [3]. Each dual-port memory block is accessed by two processors not only for data but also for task sharing. To simplify the evaluation of graceful degradation, HPEC can be considered as a degradable system without repair whose utilization period is the time between successive scheduled maintenance. For a gracefully degrading computer system without repair, the relevant performance can be measured as the total number of working interprocessor

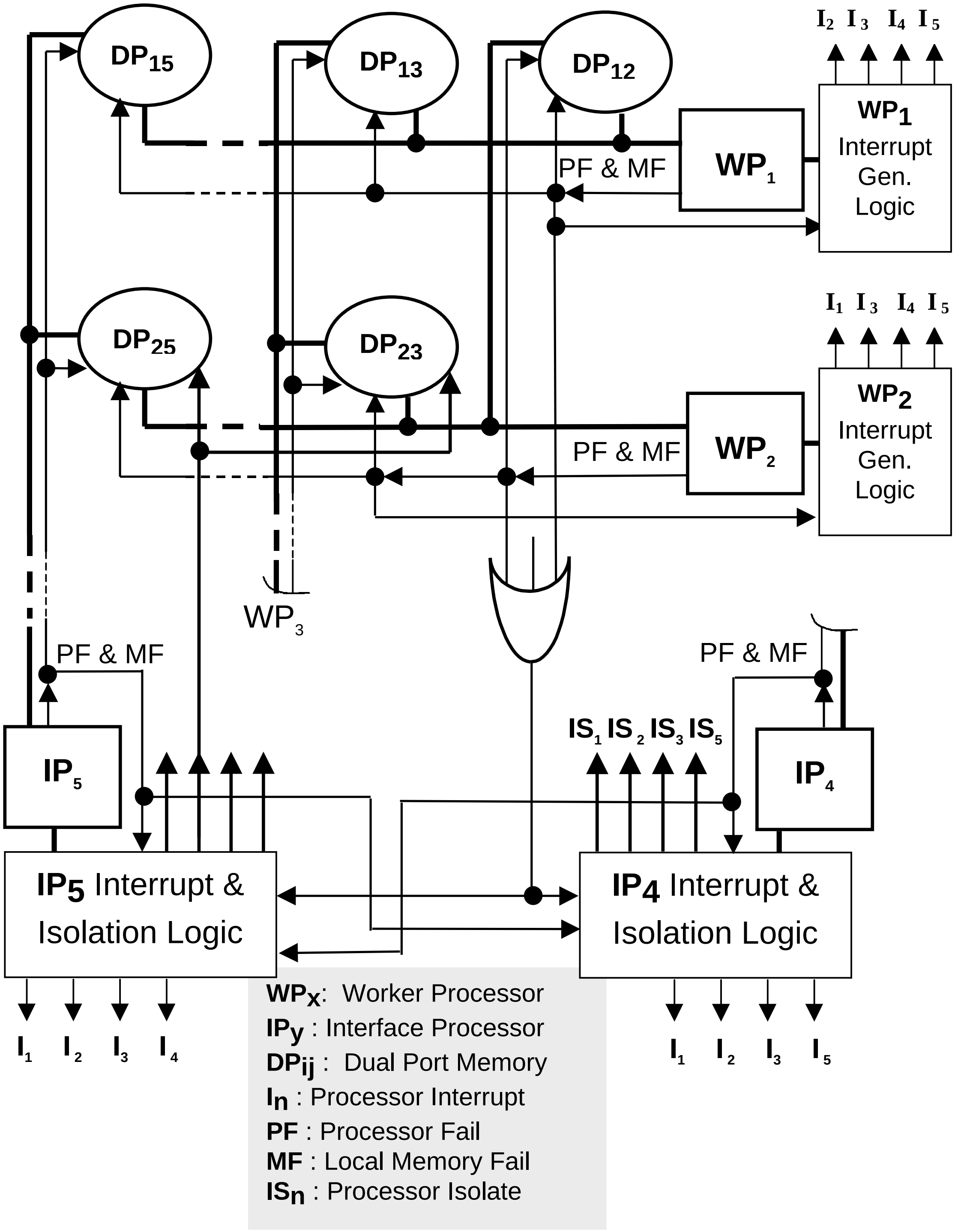

Figure 3: HPEC Fault Detection and Containment

communication paths in a given utilization time interval. An expected value of communication paths represents the expected energy that the system provides at the end of a given utilization period. Lipovski and Malek [9] employed a similar approach to evaluate multistage-network based shared memory systems. This approach avoids the complexities of enumerating all states of the system for performability evaluation. However, all possible final states of the system and their corresponding probabilities are evaluated. The performance in terms of available energy and processor interconnection paths is associated with each final state, which can be averaged over the final states.

DP memory units handle the interprocessor communication. We assume that each processing node and DP memory unit has a failure rate of λ_p and λ_m respectively (where $\lambda_p >> \lambda_m$) and their failures are independent to each other. The degradation of a five processor HPEC system without repair is analyzed in the time interval *(0, t)*. We also assume that initially all the HPEC components are operational and system fault detection and isolation capabilities are operating in the presence of faults. HPEC consists of the following main components:

- Healthy processing nodes (processor and its local memory), P = 5
- Healthy dual-port memory units
  $DP_{xy} = P(P-1)/2 = 10$

In a fault free system, the interconnection paths are equal to DP memory units. We are only considering processing nodes and DP memory units' failures. For a P processor and M dual-port memory units, the performance measure is assumed as the number of working interconnections, $\mathbf{C_{max}} = f(P, M)$.

$$f(P, M) = [P*(P-1)*(M+1)]/k \quad ..... \quad (1)$$

where k ≅ 20 for a five processor system

For ***p*** faulty processing nodes and ***m*** faulty DP memory units, the performance measure,

$$\mathbf{C} = f(P-p, M-m) = (P - p)*(P - p -1)*(M - m + 1)/k \quad ..... \quad (2)$$

The performance measure has the following properties, which were also identified by Lipovski and Malek [9].

- The performance measure, C reaches maximum for a fault free system.
- Its value decreases monotonically with the faulty components.
- It is equal to zero when all processors fail.
- Degradation due to processor failure is greater than the memory unit failure.

Normalized performance coefficient, $\mathbf{C_{nor}}$ is defined as:

$$\mathbf{C_{nor}} = f(P-p, M-m+1)/C_{max}$$
$$= [(P-p)*(P-p-1)*(M - m + 1)]/[P*(P-1)*(M+1)]$$

$\mathbf{C_{nor}}$ lies in the interval, $0 \le \mathbf{C_{nor}} \le 1$, which makes it possible to compare the graceful degradation of different size HPEC.

Considering one or a small number of node failures,

$$\mathbf{C_{nor}} = (((P-1)*(P-2p)*(M - m + 1))/(P*(P-1)*(M+1))$$
$$= ((P-2p)/P) * ((M - m + 1)/(M+1)) \quad ..... \quad (3)$$

We can also define the system reliability level R(t) in the time interval *(0, t)* as a function of expected value of performance coefficient $\mathbf{C_{nor}}$ and probabilities r(p) and r(m). The probability r(p) and r(m) are of having exactly ***p*** faulty processing nodes out of ***P*** nodes and ***m*** DP memory units out of ***M*** units.

$$R(t) = \sum_{p=0}^{P} \sum_{m=0}^{M} r(p) * r(m) * C_{nor} \quad ..... \quad (4)$$

Where r(p) and r(m) probabilities can be approximated by a Poisson distribution. If $\lambda_p t$ and $\lambda_m t$ are the probabilities of failure of each processing node and each DP memory unit respectively and $\lambda_p t$ & $\lambda_m t << 1$. Then

$$r(p) = (\lambda_p t)^p / p! * \exp(-\lambda_p t) \quad ..... \quad (5)$$
$$r(m) = (\lambda_m t)^m / m! * \exp(-\lambda_m t) \quad ..... \quad (6)$$

The seperability of the performance coefficient, $\mathbf{C_{nor}}$ results in representing the system reliability of equation (4) given below.

$$Rsys = \sum_{p} r(p)*(P-2p)/P \sum_{m} r(m)*(M-m+1)/(M+1) \quad ..... \quad (7)$$

It can be further simplified by substituting the values of r(p) and r(m) given in equation (5) and (6).

$$Rsys = \sum_{p=1}^{P} (r(p) * (1 - 2p/P)) \sum_{m=1}^{M} (r(m)*(1 - m/(M+1)))$$

The first summations $\Sigma r(p)$ and $\Sigma r(m)$ are equal to unity and second summations $\Sigma r(p)*p$ and $\Sigma r(m)*m$ represent the expected values of Poisson arrivals with an arrival rate of $P\lambda_p t$ and $M\lambda_m t$ respectively. Therefore

$$Rsys = (1 - 2\lambda_p t) * (1 - \lambda_m t) \quad ..... \quad (8)$$

Equation (8) indicates that HPEC degradation is affected more by its processing node failures than DP memory unit failures.

4.2. Reliability Modeling

HPEC system is logically divided into two non-overlapping partitions (computing and IO). The computing partition is considered operational when at least one-out of-three **WP** processors is functioning. Therefor, for a worker processing node reliability of ***Rwp*** the computing partition reliability ***Rcomp*** can be expressed as:

$$Rcomp = Rwp^3 - 3Rwp^2 + 3Rwp \quad ..... \quad (9)$$

Similarly for the IO partition, one-out of-two **IP** processors must be functioning for the proper operation of IO partition. Therefore, for interface processing node reliability ***Rip*** the IO partition reliability $\boldsymbol{R_{io}}$ is given as:

$$R_{io} = 2Rip - Rip^2 \quad ..... \quad (10)$$

Assuming that for HPEC system to be operational, one processing node in each partition and one DP memory unit connecting the two working nodes must be operational. The overall HPEC system reliability ***Rsys*** is expressed as a function of ***Rcomp*** and $\boldsymbol{R_{io}}$.

$$\begin{aligned} Rsys &= Rcomp * R_{io} \\ &= (Rwp^3 - 3Rwp^2 + 3Rwp) * (2Rip - Rip^2) \end{aligned}$$

For $Rp = Rwp = Rip$

$$Rsys = Rp^2 (6 - 9Rp + 5Rp^2 - Rp^3) \quad ..... \quad (11)$$

Nevertheless, in the worst case HPEC will continue functioning until the IO partition is operational. In other words, the system can also be functioning after the failure of computing partition. Therefore HPEC reliability can be modeled as the reliability of IO partition as:

$$Rsys = Rio = 2Rip - Rip^2$$

5. Concluding Remarks

The HPEC system can be constructed using off the shelf dual-port memories, microcontrollers and fixed-point processor cores. A VLSI implementation of HPEC is being planned. The system degrades gracefully and its design is fully modular. HPEC is considered as a gracefully degradable parallel system with no repair in a given utilization time interval for reliability analysis and evaluation. The performance measure of the system has been modeled in terms of healthy processor interconnection network (dual-port memory) paths. The modeling results indicate that the performance of HPEC degrades gracefully under various fault scenarios. The system has a high reliability and mean time to failure provided the reliable and efficient fault detection and recovery procedures are implemented. The most critical part of HPEC is the IO partition and one of the interface processors must be functioning for the operation of HPEC. Therefore a distributed recovery block scheme [7] is recommended to handle both software and hardware faults in the IO partition.

References

[1] P. G. Paulin, C. Liem, M. Cornero, F. Nacabal and G. Grossens, "Embedded software in real-time signal processing systems: application and architecture trends", *Proceedings of the IEEE*, **85**(3) 1997 pp. 419-435.

[2] R. Wilson, "Higher speeds push embedded systems to multiprocessing", *Computer Design*, July 1989 pp. 72-83.

[3] G. N. Khan, K. Mahmud, M. S. Iqbal and H. U. Rashid, "RSM - A restricted shared memory architecture for high speed interprocessor communication", *Microprocessors and Microsystems* **18**(4) 1994 pp. 193-203.

[4] D. A. Rennels, "Fault-tolerant computing--concepts and examples", *IEEE Transactions Computers* **C-33**(12) 1984 pp. 1116-1129.

[5] Russel J. Abbott, "Resourceful systems for fault tolerance, reliability and safety", *ACM Computing Survey* **22**(1) 1990 pp. 35-68.

[6] J. C. Laprie, J. Arlat, C. Beounes and K. Kanoun, "Definition and analysis of hardware- and software-fault-tolerant architectures", *IEEE Computer* **23**(7) 1990 pp. 39-61.

[7] K. H. Kim and Howard O. Welch, "Distributed Execution of Recovery Blocks: An Approach for Uniform Treatment of Hardware and Software Faults in Real-Time Applications", *IEEE Transactions Computers* **38**(5) 1989 pp. 626-636.

[8] K. Hwang and T. P. Chang, "Combinatorial reliability analysis of multiprocessor computers", *IEEE Transactions on Reliability*, **R-31**(5) 1982 pp. 469-473.

[9] G. J. Lipovski and M. Malek, *Parallel Computing, Theory and Comparisons*. John Wiley & Sons, New York 1987.