Fault-Tolerant Real-Time Scheduling Algorithm for Energy-Aware Embedded Systems

Chafik Arar, Hamoudi Kalla, Salim Kalla, Sonia Sabrina Bendib

Fault-Tolerant Real-Time Scheduling Algorithm for Energy-Aware Embedded Systems

Chafik Arar, Hamoudi Kalla, Salim Kalla and Bendib Sonia Sabrina
Department of Computer Science, University of Batna, Algeria
Email: {arar.chafik,hamoudi.kalla}@gmail.com, {salim.kalla,Bendib.SS}@univ-batna.dz

***Abstract*—In this paper, we propose a fault-tolerant scheduling approach that achieves low energy consumption and high reliability efficiency. Our scheduling solution is dedicated to multi-bus heterogeneous architectures, which take as input a given system description and a given fault hypothesis. It is based on active redundancy to mask a fixed number k of failures supported in the system, so that there is no need for detecting and handling such failures. In order to maximize the system's reliability, the replicas of each operation are scheduled on different reliable processors. Our solution can maximize reliability and reduce energy consumption when using active redundancy.**

I. Introduction

Embedded systems invade many sectors of human activity, such as medical applications, transportation, energy production, robotics, and telecommunication. The progresses achieved in electronics and data processing improves the performances of these systems. As a result, the new systems are increasingly small and fast, but also more complex and critical, and thus more sensitive to faults. The presence of some faults in these systems, accidental (design, interaction, ...) as well as intentional (human, virus, ...), are inevitable. Due to catastrophic consequences (human, ecological, and/or financial disasters) that could involve a fault, these systems must be fault-tolerant [1]. A fault can affect either the hardware or the software of the system. In this paper, we consider only processors faults in distributed architectures with buses. A bus is a multipoint connection characterized by a physical medium that connects all the processors of the architecture. As we are targeting embedded systems with limited resources (for reasons of weight, encumbrance, energy consumption, or price constraints), we investigate only software solutions.

We address hardware fault-tolerant approaches based on scheduling algorithms, to tolerate processors faults in distributed architectures. The approach that we propose is our most recent work for building fault-tolerant distributed embedded real-time systems. Prior results have been published in [2], [3]. In this paper, we are interested in approaches based on scheduling algorithms that maximize reliability and reduce energy consumption when using active redundancy to tolerate processors faults.

The paper is organized as follows. Section II describes the system model and states the faults assumptions. Section III presents our approach for providing fault-tolerance. Finally, Section IV concludes the paper.

II. Models

The algorithm is modeled as a data-flow graph, called algorithm graph and noted ALG. Each vertex of ALG is an operation and each edge is a data-dependence. A data-dependence corresponds to a data transfer between a producer operation and a consumer operation. o1 $\rightarrow$ o2 means that o1 is a predecessor of o2, and o2 is a successor of o1. Figure 1(a) presents an example of an algorithm graph, with eight operations I, I',A, B, C, D, O and O'.

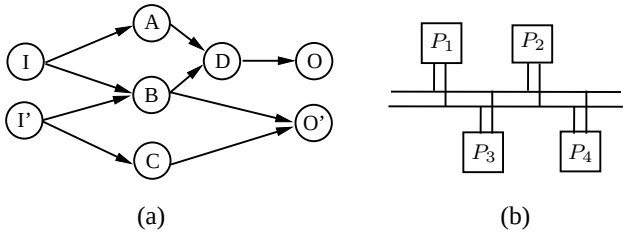

Fig. 1. Algorithm and architecture graphs.

The architecture is modeled by a non-directed graph, noted ARC, where each node is a processor, and each edge is a bus. We assume that the architecture is heterogeneous and fully connected. Figure 1(b) is an example of ARC, with four processors P_1, P_2, P_3 and P_4, and two buses Bus_1 and Bus_2.

We assume only hardware components failures and we assume that the algorithm is correct w.r.t. its specification, i.e., it has been formally validated, for instance with model checking and/or theorem proving tools. We consider only transient processors faults. We assume that at most k processors faults can arise in the system, and that the architecture includes more than k processors.

III. The proposed approach

The solution that we propose to tolerate processors faults is based on the active redundancy of operations. The advantage of the active redundancy is that the obtained schedule is static; in particular, there is no need for complex on-line re-scheduling of the operations that were executed on a processor when the latter fails; also, it can be proved that the schedule meets a required real-time constraint, both in the absence and in the presence of faults. In many embedded systems, this is mandatory.

To tolerate upto k arbitrary processors faults, each operation o of ALG is actively replicated on k+1 processors of ARC. For example, to tolerate two processors faults, three replicas of each operation of the ALG given in Figure 1(a) are scheduled on different processors (see Figure 3). We assume that all values returned by the k+1 replicas of any operation o of ALG are identical.

Figure 2 presents an example of a fault free schedule, where operations are not replicated.

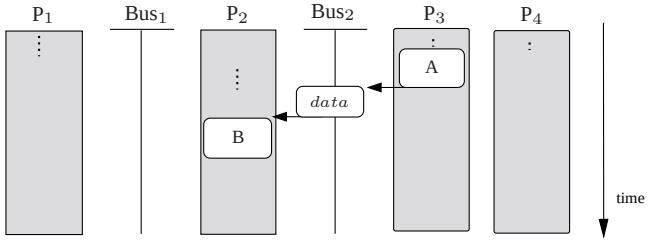

Fig. 2. A fault-free schedule.

In order to maximize the reliability R of the schedule, we propose to use the Global System Failure Rate per time unit (GSFR) function that we have proposed in [3]. The GSFR is the failure rate per time unit of the obtained multiprocessor schedule. In our approach, the replicas of each operation are scheduled on the best processors that minimizes GSFR.

The GSFR of scheduling an operation o_i, noted Λ, is computed by the following equation:

$$\Lambda(S_n) = -\log R(S_n)/U(S_n) \tag{1}$$

where, S_n is the static schedule at step n of the algorithm, and $U(S_n)$ is the total utilization of the processors.

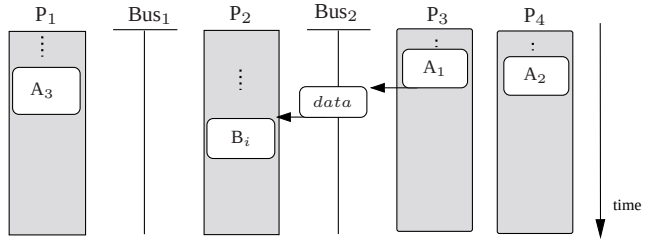

Fig. 3. A reliable fault tolerant schedule.

A. Voltage, frequency, and energy consumption

The maximum supply voltage is noted V_{max} and the corresponding highest operating frequency is noted f_{max}. For each operation, its execution time Exe assumes that the processor operates at f_{max} and V_{max}. The execution time of the operation placed onto the processor p running at frequency f (taken as a scaling factor) is:

$$Exe(X, p, f) = Exe(X, p)/f \tag{2}$$

Concerning the power consumption, we follow the model of Zhu et al. [4]. For a single operation placed onto a single processor, the power consumption Pc is:

$$Pc = P_s + h(P_{ind} + P_d) \tag{3}$$

where, $P_d = Cef\ V^2\ f$, Ps is the static power (power to maintain basic circuits and to keep the clock running), h is equal to 1 when the circuit is active and 0 when it is inactive, P_{ind} is the frequency independent active power, P_d is the frequency dependent active power, Cef is the switch capacitance, V is the supply voltage, and f is the operating frequency.

For a multiprocessor schedule S, we cannot apply directly equation 3. Instead, we must compute the total energy E(S) consumed by S, and then divide by the schedule length L(S):

$$P(S) = E(S)/L(S) \tag{4}$$

In our approach, as the k+1 replicas of each operation are scheduled actively on k+1 distinct processors, the energy consumed by the system is maximal. In order to reduce energy consumption, we propose to execute the k+1 replicas of an operation with different frequencies f. As all the k+1 replicas of each operation may have different end execution time (see Figure 3 for the replicas of A), we choose to align the execution time of all the replicas by changing the frequency f of each replica. For example, to reduce energy consumption of the system of Figure 3, we align the first two replicas A_1 and A_2 with A_3 by changing frequencies as shown in Figure 4.

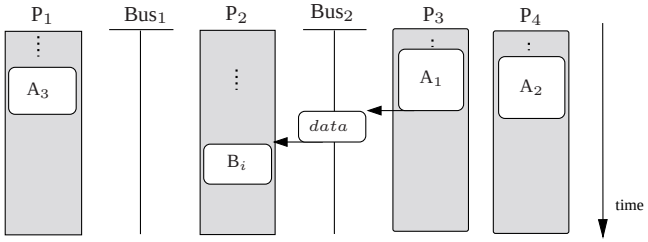

Fig. 4. The proposed scheme to reduce energy consumption

The new frequency of each replica o_i of o is depend on the end execution time of the last scheduled replica of o.

IV. Conclusion

We have proposed in this paper a solution to tolerate several processors faults in distributed heterogeneous architectures with multiple-bus topology. The proposed solution is based on active redundancy, and on GSFR to maximize reliability and to reduce energy consumption. Currently, we are working to evaluate our approach.

References

[1] N. Suri and K. Ramamritham, "Editorial: Special section on dependable real-time systems," *IEEE Trans. Parallel and Distributed Systems*, vol. 10, no. 6, pp. 529–531, Jun. 1999.

[2] I. Assayad, A. Girault, and H. Kalla, "Tradeoff exploration between reliability, power consumption, and execution time for embedded systems - the tsh tricriteria scheduling heuristic," *STTT*, vol. 15, no. 3, pp. 229–245, 2013.

[3] A. Girault and H. Kalla, "A novel bicriteria scheduling heuristics providing a guaranteed global system failure rate," *IEEE Trans. Dependable Sec. Comput.*, vol. 6, no. 4, pp. 241–254, 2009.

[4] D. Zhu, R. Melhem, and D. Mossé, "The effects of energy management on reliability in real-time embedded systems," in *International Conference on Computer Aided Design, ICCAD'04*, San Jose (CA), USA, Nov. 2004, pp. 35–40.