DAA/Langley

STATUS REPORT

on the Projects under Support of

NASA Grants NAG-1-296, NAG-1-492, and NGT-23-005-801

By

Kang G. Shin

Department of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, Michigan 48109
(313) 763-0391

Prepared for:

NASA Langley Research Center
Hampton, Virginia 23665

Attention: R. W. Butler

August 1985

1. INTRODUCTION

In this status report, we summarize the research accomplishments, under the auspices of NASA grants NAG-1-296, NAG-1-492, and NGT 23-005-801, in the area of real-time computing during the period of September 1984 - August 1985. Since real-time computing systems usually require both high performance and high reliability, our research effort has been focused on the design and analysis of computers that are fast and reliable.

2. SUMMARY OF ACCOMPLISHMENTS

Due to the long-term nature of our projects, we have taken a step-by-step approach to the design and analysis of real-time computing systems, beginning from the definition of new performance measures and analytical modeling/simulation to experimental validations. Validations have been pursued mainly via various experiments at the NASA AIRLAB. Two of the three such Airlab experiments that we have conducted during this period are outlined below.

Application of our performance measures to the design and evaluation of real-time, fault-tolerant computers has yielded several very interesting results as shown below and elsewhere.

2.1. Experiments on FTMP at the NASA Airlab

Our fundamental approach consists of both analytical modeling and experimental validation. Although the analytical part has been far more advanced than the experimental counterpart (for both our case and others'), we feel that experimental validation is *difficult* but *essential*.

To this end, three experiments on FTMP were begun, and one of them has been completed, and the remaining two are now in progress. These are (i) measurement of fault latency in FTMP [13],[16], (ii) validation and analysis of FTMP synchronization protocols [11],[16], and (iii) investigation of error propagation in FTMP. (i) has been completed and (ii) and (iii) are now in progress. Experiments (ii) and (iii) are described below. (See [13], [16] for a detailed description of (i).)

2.1.1. Evaluation of FTMP Sy onization

We have developed an analytic model that represents the state of a processing element in a multiprocessor with time-shared buses. This model consists of several states representing idle, processing, failed, communicating, and waiting states of a single processor or a single triad of processors. Especially, the model emphasizes (i) validation of processor synchronization protocols and (ii) the impacts of bus contention and failure handling on computer performance, i.e. dynamic failure probability, and mean and variance costs.

The transition rates between the model states will describe the effects of workload, bus contention, architecture, and reliability on a multiprocessor system. To measure these transition rates, we have adapted our model to the architecture of FTMP and have already conducted some experiments on FTMP at the NASA AIRLAB. During our recent trips to the AIRLAB, we have been able to make direct measurements on the FTMP to aid in arriving at experimental values for the transition rates. Measurements are made through software functions and using the logic analyzer in the Lab. The measurements obtained so far are preliminary. To obtain more concrete measurements, experiments at Airlab are planned to continue in the coming months and years.

With the results of FTMP experiments, we will have a better understanding of the accuracy of the model and how it may be solved analytically. The validated model will then be used as a dependable but economical tool for evaluation and design of real-time avionics computers.

2.1.2. Measurement of Error Propagation in FTMP

The most important parameter in our error propagation model (discussed in our last year's proposal) is the error propagation time for each unit. When a fault occurs in the system, it will induce an error first in the faulty unit, and this error may propagate and cause additional errors in other units prior to its detection. (This is analogous to "snowballing effects".) In such a case, the error is said to propagate from a *faulty point* to a *detection point*. If the system is divided into many units, each of which may have inputs from several other units and outputs to several other units, then the error propagation time is defined for each input/output pair of a unit as the time for an error to propagate from an input port to an output port. The purpose of our current experiments at the NASA AIRLAB is to directly measure error propagation times on FTMP.

The processor region of an LRU in FTMP can be divided into the following units: (1) a CAPS-6 bit-sliced microprocessor, (2) a cache memory including 8K PROM and 8K RAM, (3) a system bus coupler, (4) a group of control and communication registers, and (5) an interval timer and an address mapper. Basically, all units communicate via the transfer bus, but only the CAPS-6 and the system bus coupler can initiate a bus transfer. Since the error detection is built upon the system bus, any error in the processor region must propagate through the system bus coupler before the system detects that error. Thus, we have to first measure the error propagation time for the system bus coupler with inputs from the CAPS-6 and outputs to the system bus interface unit.

The method we use to detect input and output errors is to compare the input and output line values of the system bus coupler between LRU3 (upon which faults are injected) and LRU0 (which should be in the same triad as LRU3 when faults are injected). By inspecting the circuit diagram, we have defined 26 input lines and 4 output lines for our experiments. We are now building the comparison circuit for error detection, and because the clocks in both LRUs are not exactly synchronized, we have encountered some difficulties in designing the circuit. However, we believe that by carefully sampling and buffering the input and output lines of the comparison circuit, it will be possible to detect errors correctly.

These experiments are expected to be continued for the rest of this year and also for the coming years. As described above, we will begin experiments at low level modules in FTMP and then expand them to higher level subsystems in FTMP as well as other systems that will be available at the AIRLAB and elsewhere.

2.2. Application of Performance Measures

We have applied our performance measures to various problems associated with the design and analysis of fault-tolerant, real-time computing systems. As can be seen below and from our publication list, there are numerous important applications, indicating the power and usefulness of our performance measures. We feel that these measures can be applied to almost all, if not all, problems associated with design, validation, and analysis of real-time computing systems. Consequently, the applications that we have dealt with thus far are only a beginning. More applications will be discussed in the forthcoming renewal proposal.

2.2.1. Optimal Reconfiguration of a Multi-Module System

We have developed a new quantitative approach to the problem of reconfiguring a degradable multi-module system. The approach is concerned with both assigning some modules for computation and arranging others for reliability.

Conventionally, a fault-tolerant system performs reconfiguration only upon a subsystem failure. Since there exists an inherent tradeoff between the computation capacity and fault-tolerance of a multi-module computing system, the conventional approach is a passive action and does not yield a configuration which provides an optimal compromise for the tradeoff. Using the expected total reward as the optimal criterion, we have shown the need and existence of an active reconfiguration strategy under which the system reconfigures itself on the basis of not only the *occurrence of a failure* but also the *progression of the mission.*

Following the problem formulation, we have investigated some important properties of an optimal reconfiguration strategy which specify (i) the times at which the system should undergo reconfiguration, and (ii) the configurations that the system should change to. Then, the optimal reconfiguration problem is converted to integer nonlinear knapsack and fractional programming problems. We have developed various algorithms for solving these problem and worked out a demonstrative example. See [7] for a detailed description.

2.2.2. Scheduling with a Quick Recovery from Failure

Multiprocessors used in life-critical real-time systems must recover quickly from failure. Part of this recovery consists of switching to a new task schedule that ensures that hard deadlines for critical tasks continue to be met. We have developed a dynamic programming algorithm that ensures that backup, or contingency, schedules can be efficiently embedded within the original, "primary" schedule to ensure that hard deadlines continue to be met in the face of up to a given maximum number of processor failures. Several illustrative examples have also been worked out. See [4] for more details.

2.2.3. Modeling of Real-Time Multiprocessors with Time-Shared Buses

In this project, the workload effects on computer performance are addressed for a highly reliable unibus multiprocessor used in real-time control [11],[16]. Because of the strict performance criteria required by a system of this type, it would be desirable to be able to determine the significant effects of workload distribution and scheduling on performance.

As an approach to studying these effects, a modified *stochastic Petri net* (SPN) is used to describe the synchronous operation of this system. From this model the vital components affecting performance can be determined. However, because of the complexity in solving the modified SPN, a simpler model is constructed that presents the same critical aspects. This model is a closed queueing network. It consists of multiserver nodes and a non-preemptive priority queue. The use of this model for a specific application requires the partitioning of the workload into *job classes*. The steady state solution of the queueing model directly produces useful results, such as idle processing time, system bus contention, and bus queueing times, that are necessary in any performance evaluation. This model has been used in evaluating FTMP.

2.2.4. Optimal Checkpointing in Real-Time Systems

Using the basic concept of checkpointing in database systems, we have developed analytical models for the design and evaluation of checkpointing in real-time systems, which usually require high performance and reliability [12].

First, we have modeled the behavior of a real-time task under the common assumption of *perfect coverage* of on-line detection mechanisms (which is termed a *basic model*). Then, we have generalized the model (to an *extended model*) to include more realistic cases, i.e., *imperfect coverages* of on-line detection mechanisms and acceptance tests. Finally, we have determined an optimal placement of checkpoints to minimize the mean task execution time while the *probability of unreliable results* (or *lack of confidence*) is kept below the specified level. Consideration of both the case of imperfect coverages and the probability of lack of confidence is a significant departure from previous approaches by others.

For the basic model we have shown that an equal distant interval is optimal, whereas for the extended model this is not necessarily true. For the latter we have derived a numerical algorithm which produces a better solution than the common usual solution, i.e., equi-distance (inter-checkpoint) intervals.

2.2.5. Synchronization of a Large Clock Network

Clock Synchronization is one of the main problems associated with the design of a multiprocessor system, especially when there exist malicious faults. Although over the past few years many different algorithms have been proposed for overcoming this problem, they are not suitable for a large real-time multiprocessor system due to their excessive time overhead and/or large number of interconnections.

To remedy this problem, we have developed a new method that (i) requires little time overhead by using the phase-locked clock synchronization, and (ii) uses only 20-30% of the total number of interconnections required by the other methods for almost no loss in the synchronizing capabilities [10]. This drastic reduction in the total number of interconnections is made possible by grouping the various clocks in the system into many different clusters and then treating the clusters themselves as single clock units as far as the network is concerned. The method is significant in that regardless of their size multiprocessor systems can be built at an inexpensive cost without sacrificing both the synchronization and fault tolerance capabilities.

To show the feasibility of our method, we have also developed an example hardware implementation.

2.2.6. Processor Tradeoff in Distributed Real-Time Systems

Optimizing the design of real-time distributed systems is important since the systems are frequently critical to life. This optimization is a difficult problem, and heuristics and designer judgement are called for in the process. The chief cause of the difficulty is the large number of parameters under the designer's control which impact performance and life-cycle cost. We have studied the interplay between the more important parameters in this project using two objective measures, i.e., the *mean cost* and the *probability of dynamic failure* that we have previously developed. Among these parameters are the processor burn-in time and processor replacement policy. A central feature of this work is a look at how the application requirements affect the optimality of the distributed systems: indeed, the application requirements are an integral part of the analysis. See [9] for more details.

2.2.7. Implications of the Interactive Convergence Algorithm

Fault-tolerant synchronization is crucial to the proper functioning of controller computers. In this project, we have assessed the *Interactive Convergence Algorithm* of Lamport and Melliar-Smith, and studied the interaction of tightness, fault tolerance, and overhead in the context of the SIFT aircraft control system. See [14] for a detailed description.

3. PARTICIPATING PERSONNEL

- Yann-Hang Lee: Ph. D. completed in December 1984. Currently working as a researcher at IBM Thomas J. Watson Research Center.
- Michael Woodbury: M. S. completed in December 1984. Currently working towards the Ph. D. degree.
- Michael Lin: Second-year Ph. D. student.
- R. Parameswaran: M. S. expected in April 1986.
- James Dolter: M. S. expected in December 1985.
- Y. Muthuswamy: New M. S. student beginning Fall 1985.

4. TECHNICAL PAPERS AND REPORTS

The following technical papers/reports present in some detail the work accomplished thus far.

[1] C. M. Krishna, K. G. Shin, and Y. - H. Lee, "On Optimal Criteria for Checkpoints Placement," Communications of ACM, Vol. 27, No. 10, pp. 1008-1012, October 1984.

[2] K. G. Shin and Y. - H. Lee, "Analysis of Backward Error Recovery for Concurrent Processes with Recovery Blocks", *IEEE Trans. on Software Engineering,* Vol. SE-10, No. 6, pp. 692-700, November 1984.

[3] K. G. Shin, C. M. Krishna, and Y. - H. Lee, "A Unified Method for Evaluating Real-Time Computer Controllers and Its Application," *IEEE Trans. on Automatic Control,* Vol. AC-30, No. 4, pp. 357-366, April 1985.

[4] K. G. Shin and C. M. Krishna, "The Processor Number-Power Tradeoff in a Class of Multiprocessors," *Proc. of the 5-th Int'l Conf. on Distributed Computing Systems,* pp. 321-328, May 1985.

[5] C. M. Krishna and K. G. Shin, "On Scheduling Tasks with a Quick Recovery from Failure", *Digest of FTCS-15*, pp. 234-239, June 1985. Also to appear in *IEEE Trans. on Computers*.

[6] C. M. Krishna, K. G. Shin, and R. W. Butler, "Ensuring Fault-Tolerance of Phase-Locked Clocks", *IEEE Trans. on Computers*, Vol. C-34, No. 8, pp. 752-756, August 1985.

[7] Y. - H. Lee and K. G. Shin, "Optimal Reconfiguration Strategies for a Degradable Multi-Module Computing System," accepted to *J. of the ACM* subject to a minor revision.

[8] Y. - H. Lee and K. G. Shin, "Optimal Design and Use of Retry in Fault Tolerant Real-Time Computer Systems", submitted to *J. of ACM*. (First revision February 1985.)

[9] C. M. Krishna, K. G. Shin, and I. S. Bhandari, "Processor Tradeoff in Distributed Real-Time Systems," to appear in *IEEE Trans. on Computers*.

[10] K. G. Shin and R. Parameswaran, "Synchronization of a Large Clock Network in the Presence" of Malicious Failures," *Proc. 1985 Real-Time Systems Symp.* (to appear). Also submitted to *IEEE Trans. on Computers*.

[11] M. H. Woodbury and K. G. Shin, "Performance Modeling of Real-Time Multiprocessors with Time-Shared Buses," submitted to *IEEE Trans. on Computers*.

[12] K. G. Shin, T. - H. Lin, and Y. - H. Lee, "Optimal Checkpointing in Real-Time Systems," submitted to *IEEE Trans. on Computers*.

[13] K. G. Shin and Y. - H. Lee, "Measurement of Application of Fault Latency," submitted to *IEEE Trans. on Computers*.

[14] C. M. Krishna and K. G. Shin, "Operational Implications of the Interactive Convergence" Algorithm," submitted to *IEEE Trans. on Software Engineering*.

[15] C. M. Krishna and K. G. Shin, "Performance Measures for Control Computers," submitted to *IEEE Trans. on Computers*.

[16] K. G. Shin, M. H. Woodbury, and Y. - H. Lee, "Modeling and Measurement of Fault-Tolerant Multiprocessors," *NASA Contractor's Report*, May 1985.

[17] Y. - H. Lee, "Characterization of Failure Handling in Fault-Tolerant Multiprocessor Systems," Ph.D. Thesis, The University of Michigan, November 1984.