Chalmers University of Technology
Department of Computer Engineering
412 96 Göteborg

1999-08-10

Node-level Fault Tolerance for Fixed Priority Scheduling

An ARTES supported research project

Project plan

Johan Karlsson

Summary

Techniques for error detection and fault tolerance for distributed real-time systems using fixed priority preemptive scheduling will be studied. Specifically, techniques for achieving node-level transient fault tolerance using massive time redundancy will be developed.

Error detection and fault tolerance mechanisms will be implemented in a small real-time kernel and evaluated using both VHDL-based fault injection and fault injection in a real system. The real-time kernel will be implemented for the THOR microprocessor developed by Saab Ericsson Space AB, who will supply a detailed VHDL simulation model of THOR and computer boards.

1 Problem statement

Embedded hard real-time systems are often used in application where system failures may pose a threat to human lives or cause significant economic loss. Examples of such applications are systems that control cars, trucks, trains, aircraft, satellites and satellite launchers. Real-time computer systems used in such application are often distributed and fault-tolerant. Fault-tolerance is imperative to meet stringent requirements on reliability and safety. Traditionally, static scheduling is used in critical application because of its robustness and that timing analysis is easy.

In modern control systems it is often desirable to use a more flexible scheduling technique to provide a higher degree of flexibility so that the system can efficiently respond to a variety of events in the environment. At the same time it is important that fault-tolerance can be implemented in a cost-effective way. Protocols used in distributed fault-tolerant real-time systems must be based on assumptions about the failure semantics of the computer nodes. One example is the *fail-silent property,* which implies that a node must either produce correct results or no results at all.

To ensure that specified failure semantics are achieved, it is important to provide the computer nodes with efficient fault handling mechanisms. In this context, an important measure is the *fault coverage,* which is defined as the conditional probability that the specified failure semantics will be fulfilled given that a fault has occurred in the computer node.

An important problem for system designers is to achieve high fault coverage at low cost. One

way to achieve high coverage for hardware faults is to use duplicated hardware and comparison internally in the node, but this solution is often considered too expensive even for safety-critical systems. A more cost-effective solution is to combine the hardware error detection mechanisms included in most modern microprocessors with software checks and time redundant execution of tasks.

Time redundant execution of tasks has long been recognized as a cost-effective way to detect transient faults in computer systems. Rece nt developments in microprocessor technology has shown tremendous increase in processing power, which has made time redundancy increasingly attractive for the design of fault-tolerant real-time systems. With the use of high-performance microprocessors, it is now feasible in many applications to use time redundancy on a massive scale to achieve not only error detection both also transient fault tolerance at the node level in a distributed system.

Most distributed fault-tolerant real-time systems provide fault-tolerance at the system-level. For example, tasks may be executed on two nodes. If one node fails, the other node still produce correct results which ensures continuity of service. However, research has shown that most computer failures are transient in their nature. Possible causes of transient faults are high-energy neutrons in aircraft, cosmic heavy-ion radiation in space applications, and power fluctuations or electromagnetic interference (EMI) in cars, trains and industrial plants. Many software faults are also appear as transient faults since software problems often can be handled by simply restarting the program. Most transient faults can be handled at the node level by combining time redundancy with behaviour-based error detection in hardware and software. If most faults are handled at the node level, it may be possible to reduce the cost of fault-tolerance at the system level. For example, instead of using active replication of tasks it may become sufficient to use a primary-backup approach. Node-level fault-tolerance could also be a cost-effective way of improving dependability of distributed real-time systems that lack system-level support for fault-tolerance.

2 Main research ideas

We propose an investigation of techniques for achieving node-level fault-tolerance and ensuring node-failure semantics when using fixed-priority scheduling in distributed real-time systems.

Our plan is to implement a small real-time kernel that uses fixed priority scheduling, and to include mechanisms for node-level error detection and fault-tolerance. The real-time kernel will be developed for the THOR microprocessor, which is manufactured by Saab Ericsson Space AB.

We have previously evaluated the node-level error detection mechanisms used in the distributed real-time system MARS [1]. The results of this study showed a detection coverage of 99.5% for heavy-ion induced transients faults when using double execution of tasks and result comparison as the highest level of error detection for a control application. This error detection coverage may not be sufficient for critical applications. We will therefore investigate more sophisticated ways of using time redundancy. We have chosen to study fixed-priority scheduling because its use in fault-tolerant systems has not been researched extensively, in particular, with respect to estimating the coverage of error detection and fault-tolerance mechanisms.

The objective is to achieve node-level fault-tolerance for transient hardware faults, and software faults in application tasks. We will consider techniques such as software assertions, end-to-end message checksums, and time redundant execution to achieve error detection. These software techniques will complement the hardware error detection mechanisms in the THOR microprocessor. To achieve fault-tolerance we will consider time-redundant execution, roll-back checkpointing and roll-forward checkpointing. We will also investigate the possibility to

use time redundancy for the execution of kernel code, as well as the impact on response time imposed by roll-back and roll-forward actions. For faults which cannot be tolerated at the node level, it is necessary that specified node-failure semantics, e.g. fail-silence, are fulfilled.

To evaluate the effectiveness of the fault-tolerance and error detection techniques, a detailed VHDL simulation model of a computer node will be developed. The main part of the simulation model is a detailed VHDL model of the THOR microprocessor provided by Saab Ericsson Space AB. Fault injection experiments will be carried out with the simulation model using MEFISTO [2], a VHDL-based fault injection tool developed by our research group. We will also evaluate the fault-tolerant real-time kernel by injecting faults into a real THOR-processor via its scan-chains using the FIMBUL tool [3], which also was developed by our research group. These two fault injection tools complement each other as MEFISTO provides a higher degree of observability and controllability, while FIMBUL allows the THOR-processor to execute at full speed.

After the investigation of node-level fault tolerance techniques, we will focus on the problem of achieving system-level fault-tolerance for a distributed system that use fixed-priority scheduling. We will extend the real-time kernel to support system-level fault-tolerance in a distributed system. We will also implement a small distributed system consisting of two nodes, and conduct fault injection experiments with this system to evaluate fault coverage and to study how fault recovery affect response time. If possible, we will also develop an analytical model for response time analysis of fixed-priority scheduling taking into account node-level and system-level fault recovery.

3 Expected results

The expected results are:

- New techniques for achieving node-level fault tolerance for systems using fixed-priority scheduling.
- New techniques for achieving system-level fault tolerance for systems using fixed priority scheduling.
- New techniques for response time analysis for fixed-priority scheduling taking into account system-level and node-level fault-tolerance.
- Insights into the problem of estimating error detection and failure assumption coverage using scan-chain implemented and simulation based fault injection.

4 Work plan

We have received funding for one student for 2 years, and plan to apply for funding for another 2 years. The expected time to achieve the Ph.D. degree is five years, since the student will work 20% as teaching assistant.

Activities

Year 1

Study of real-time scheduling and fault-tolerance techniques.

Study of the THOR-processor and the fault injections tools (FIMBUL and MEFISTO)

Implementation of a first version of a real-time kernel with node-level fault-tolerance

Fault injection experiments to evaluate the first version of the real-time kernel using the MEFISTO and/or FIMBUL. (Which tool to use first will be decided during the project).

Year 2 and 3

Based on the results of the fault injection experiments, a refined version of the real-time kernel is developed.

The new version of the real-time kernel is evaluated using MEFISTO and FIMBUL.

Defence of licentiate thesis after 2,5 years.

Study of techniques for achieving fault-tolerance in distributed real-time systems using fixed-priority scheduling

Year 4 and 5

Implementation of a real-time kernel that support fault-tolerance for distributed fault-tolerant computing using fixed-priority scheduling.

Implementation of a small distributed system consisting of two nodes.

Fault injection experiments to validate system-level as well as node-level fault-tolerance features of the small distributed system.

Development of analytical techniques for response time analysis for fixed-priority scheduling taking into account system-level and node-level fault-tolerance.

Defence of Ph.D. degree.

Deliverables

Year 1

Technical report describing the real-time kernel

Year 2-5

One conference paper each year

Year 3

Licentiate thesis

Year 5

Ph D thesis

5 Related research

Our earlier evaluation of the MARS system is described in [1]. The MEFISTO tool is described in [2], and the FIMBUL tool in [3]. Fixed-priority scheduling has been described in many research papers and text books such as [4]. Roll-forward and roll-back checkpointing techniques are described in [5]. Examples of fault injection based evaluation can be found in [6] and [7].

6 Relation to profile

This project investigates software implemented techniques for error detection and fault-tolerance that make it possible construct fault-tolerant real-time systems using COTS (commercial-off-the-shelf) microprocessors and hardware components. Traditionally, tailor made hardware solutions has been used to achieve high fault coverage in fault tolerant systems. Thus, the project address the goal of designing real-time systems using non-tailor made components. The projects relation to dependability and distributed systems is obvious.

7 Industrial relevance

The project will be carried out in close cooperation with Saab Ericsson Space AB, which will provide simulation models and computer boards based on the THOR-microprocessor. The results of the study is expected to be of significant value to Saab Ericsson Space for future development of the THOR-processor and on-board computer systems for launchers and satellites.

8 Relation to other SSF programmes

The research group has currently no funding from other SSF programmes.

9 Context

The project is conducted within the Laboratory for Dependable Computing (LDC) at Chalmers University of Technology. Head of LDC is Professor Jan Torin. Project leader will be Associate Professor Johan Karlsson, Institutionen för datorteknik, Chalmers tekniska högskola, 412 96, Göteborg, tel 031 - 772 16 70.

The work is carried out in a research group focusing on validation of fault-tolerant systems led by Johan Karlsson. Currently the group consists of four Ph D students. The group is supported by grants from NUTEK and NFFP (Nationella flygforskningsprogrammet).

Contact at Saab Ericsson Space AB is Torbjörn Hult, Saab Ericsson Space AB, Delsjömotet, 405 15 Göteborg, tel 031 - 735 00 00.

10 References

[1] Johan Karlsson, Peter Folkesson, Jean Arlat, Yves Crouzet, Günther Leber and Johannes Reisinger, “Application of Three Fault Injection Techniques to the Experimental Assessment of the MARS Architecture”, *Proc. 5th IFIP International Working Conference on Dependable Computing for Critical Applications (DCCA-5*), IEEE Computer Society Press, Urbana, IL, USA, September 1995.

[2] Eric Jenn, Jean Arlat, Marcus Rimén, Joakim Ohlsson and Johan Karlsson, “Fault Injection into VHDL Models: The MEFISTO Tool”, *24th Int. Symp. on Fault Tolerant Computing (FTCS-24)*, IEEE, Austin, TX, USA, June 1994.

[3] Peter Folkesson, Sven Svensson and Johan Karlsson, “A Comparison of Simulation Based and Scan Chain Implemented Fault Injection”, *Proc. 28th Int. Symp. on Fault Tolerant Computing (FTCS-28)*, IEEE Computer Society Press, Munich, Germany, June 1998

[4] C.M. Krishna, Kang G. Shin, *Real-time systems*, McGraw-Hill, 1997, ISBN 0-07-114243-6.

[5] D.K. Pradhan, *Fault-Tolerant Computer System Design*, Prentice Hall, 1996, ISBN 0-13-057887-8.

[6] Marcus Rimén, Joakim Ohlsson and Johan Karlsson, “Experimental Evaluation of Control Flow Errors”, *Proc. 1995 Pacific Rim International Symposium on Fault Tolerant Systems (PRFTS)*, IEEE Computer Society Press, Newport Beach, CA, USA, December 1995.

[7] Ghassem Miremadi, Joakim Ohlsson, Marcus Rimén and Johan Karlsson, “Use of Time and Address Signatures for Control Flow Checking”, *Proc. of the 5th International Working Conference on Dependable Computing for Critical Applications (DCCA-5)*, IEEE Computer Society Press, Urbana Champaign, IL, USA, September 1995