Time-Predictable Fault Tolerant Computing for Dependable Automotive Cyber-Physical Systems

Wei Zhang
Associate Professor
Department of Electrical and Computer Engineering
Virginia Commonwealth University
wzhang@vcu.edu

Dependable and secure automotive cyber-physical systems (CPSs) are crucial as human's lives are dependent on them. Many important subsystems in today's automobiles such as the engine control system and the anti-brake system are hard real-time systems. If the CPUs in those systems have any fault, regardless of transient faults or hard faults, not only the computation results may be wrong, but also the results may be delivered late. Therefore, CPUs used in those systems must be able to handle two tasks: 1) detect and correct the errors, and 2) ensure that the error detection and correction can be done within the deadline so that the system can function correctly or have a grace period.

Microprocessors have been deeply embedded in today's automobiles. However, due to the advances of computer architectures that are generally focused on improving the average-case performance, many architectural features such as caches and pipelines have made it very hard to safely and accurately predict the worst-case execution time (WCET), which is crucial for ensuring schedulability of hard real-time tasks. Therefore, in light of the dependability against possible transient and hard errors, the CPU itself must be both high performance and time predictable in the fault-free case. Then on top of that, we can study fault-tolerant techniques to maintain the time predictability. Therefore, we propose to study time-predictable fault tolerant techniques based on the Real-time Very Long Instruction Word (RVLIW) processor we designed, which can provide both high performance and time predictability by leveraging the static scheduling and compiler optimizations [1].

1. Real-Time Fault Tolerance Design based on RVLIW

Due to the scaling of technology, modern microprocessors are subject to both transient errors (i.e. soft errors or single-event upsets) and hard errors. While there have been many research efforts on protecting processors from both transient and hard errors, few work has considered how to tightly bound and accurately estimate the time to detect and correct soft/hard errors, which is crucial for hard real-time systems. To ensure both reliability and time predictability, we propose the following framework as depicted in Figure 1.

The processor is based on RVLIW, which as *n* functional units, and *n* is a multiple of 3. While N-Modular Redundancy (NMR) is known as a highly reliable yet expensive technique, due to the abundance of functional units and other resources in a modern VLIW processor, it becomes cost-effective to implement software-based NMR. Specifically, we propose to replicate the software task (i.e. T) two times, i.e. T' and T'' and schedule them on three different functional units. Of course, each task can run on more than 1 functional units, if *n* is larger than 3. Then the results of all three threads will be compared at critical points to determine whether or not there is any fault. If there is, the majority voting can be simply used to choose the right value for quick recovery. It should be noted that this technique can identify both soft and hard errors, because if a functional unit repeatedly reports errors, it can be identified as having a hard fault and should be bypassed in future computing. This computing platform will be incorporated and evaluated in the CPS testbed we described in Section 2.

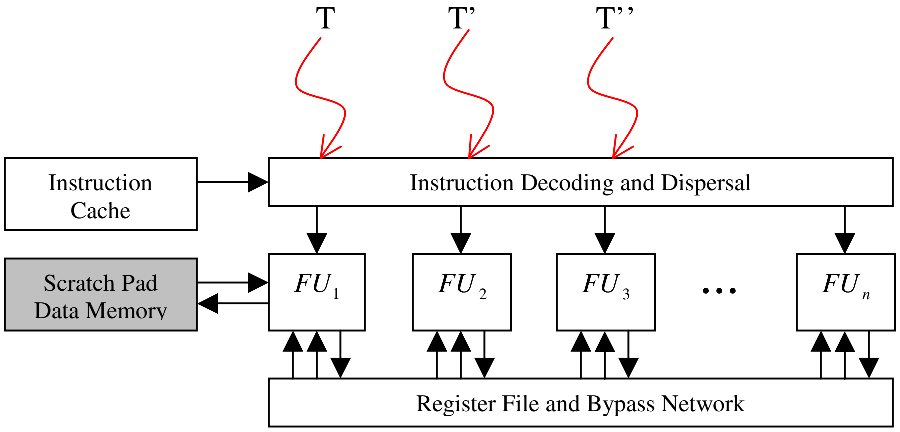

Figure 1. Real-time fault tolerant computing framework based on RVLIW

The research problems include the following:

1. How often we need to do comparison so that we can detect the both soft and hard errors promptly?
2. What is the impact on the execution time of T if it is replicated two times and run concurrently with its two duplicates T' and T''?
3. How can we optimize the instruction scheduling, code and data layout to exploit the locality for reducing the performance overhead of the software-based 3 modular redundancy?
4. How do we customize the instruction cache and scratchpad memory to ensure both time predictability and good memory performance?
5. What will be the worst-case execution time by considering both the error detection and recovery time?

2. An Automotive CPS Testbed for Research and Education

Another important project our group plans to conduct is to develop a cost-effective CPS testbed that can be used for both research and teaching. I believe a significant barrier for advancing CPS research today is the lack of an affordable CPS testbed, without which most researchers are limited to their traditional and "comfortable" research areas and it is hard to engage researchers in other relevant fields to collectively address critical challenges of CPSs.

We are inspired by the work at the convergence laboratory at the University of Illinois [2]. We plan to build an open testbed consisting of networked remotely controlled cars based on mostly off-the-shelf components. Each car will employ a set of sensors (i.e., touch and IR sensors) and a controller based on Model Predictive Control (MPC), and a wireless ad-hoc network will be used to support the communication between the remote laptops and cars. It would also be interesting to develop an open-source hardware-in-the-loop simulator as well for certain subsystem of automobiles, for example the brake control system, so that we can easily evaluate the performance and time predictability of our real-time fault-tolerant processor with simulated error injection. The cost of such an open testbed should be within about $1-2K to ensure affordability and reusability, although the number of cars should be made scalable to support various studies with different levels of complexity.

Reference:

[1] J. Yan, W. Zhang. A time-predictable VLIW processor and its compiler support. In the Journal of Real-Time Systems, Vol. 38, No. 1, pp. 67 – 84, Jan. 2008.
[2] https://netfiles.uiuc.edu/prkumar/www/testbed/videoclips.html.
[3] T. Hwang, J. Rohl, K. Park, J. Hwang, K. H. Lee, K. Lee, S.-J. Lee, and Y.-J. Kim, "Development of HIL Systems for active Brake Control Systems", SICE-ICASE International Joint Conference, 2006.

Bio of the author **Wei Zhang**: Dr. Wei Zhang is an associate professor in Electrical and Computer Engineering at Virginia Commonwealth University. He received his Ph.D. degree in computer science and engineering from the Pennsylvania State University in 2003. Since then, he has worked as an assistant and an associate professor at Southern Illinois University Carbondale until August 2010. His research interests are in embedded and real-time computing systems, computer architecture and compiler. Dr. Zhang has received the 2009 SIUC Excellence through Commitment Outstanding Scholar Award for the College of Engineering, and 2007 IBM Real-time Innovation Award. His research has been supported by NSF, IBM and Altera