Challenges for Cyber-Physical Systems: Security, Timing Analysis and Soft Error Protection

Frank Mueller
Department of Computer Science, North Carolina State University, Raleigh, NC, *mueller@cs.ncsu.edu*

Abstract—**This paper highlights multiple shortcomings in the current design process of cyber-physical embedded systems with real-time constraints.**

First, shortcomings in current as well as future standards to controlling the power grid are outlined. From these economic and safety threats, we derive an immediate need to invest in research on the protection of the power grid, both from the perspective of cyber attacks and distributed control system problems.

Second, current software design practice does not adequately verify and validate worst-case timing scenarios that have to be guaranteed in order to meet deadlines in safety-critical embedded systems. This equally applies to avionics and the automotive industry, both of which are increasingly requiring their suppliers to provide verifiable bounds on worst-case execution time of software. Yet, there is a lack of viable solutions that suppliers can employ. We provide an analysis of this problem that outlines directions for future research and tool development in this area, both of which are pressing issues.

Third, the correctness of embedded systems is currently jeopardized by soft errors that may render control systems inoperable. In general, soft errors are increasingly a problem due to (a) smaller fabrication sizes and (b) deployment in harsh environments. Increasingly, off-the-shelf embedded processors without hardware protection against soft errors are being deployed in airplanes and cars. Meanwhile, system developers have been asked to consider the effect of soft errors in their software design, yet they lack a methodology to do so. We outline much needed research in this area.

I. Security Concerns in the Power Grid

The power grid represents a distributed cyber-physical system that is essential to our every-day life. Larger-scale black-outs are known to have a severe economic and safety impact, as historical events have shown. The severity in impact of power outage on our life is increasing continuously as the power distribution grid becomes more standardized and more automated. Current standardization efforts include the forthcoming IEC 61850 protocol that will eventually replace existing DNP variants and other protocols. The 61850 standard redefines the interaction between substations that provide power to, *e.g.*, quarters of a city and control centers that coordinate power distribution to balance supply and demand. This includes an increasing trend in substation automation, mainly to increase efficiency and reduce maintenance overhead. However, substation automation poses a potential for power outages should they become the target of cyber attacks or should a distributed control system malfunction. The effects could be as small as long-lasting blackouts for regions serviced by the substation or as large as larger-scale blackouts if damage is inflicted in an orchestrated, distributed attack or cascades for technical reasons.

Current DNP and future 61850 standards are deployed over regular Ethernet. The long-haul connections to control centers are typically dedicated lines and, hence, are considered safe from cyber attacks. While this assumption may not be sound, substations themselves are a more likely target as they are unmanned. Physical access within a substation (or via local wireless maintenance link at a substation) could allow attackers to affect power devices. Some protection could be provided by current systems, such as encryption at the TCP layer given that DNP and 61850 traffic is, in large, layered over TCP in practice. However, some messages have real-time requirements, which cannot be guaranteed by TCP. These messages remain extremely vulnerable to attacks as they cannot easily be encrypted given that packet transmission occurs at the link layer in current solutions. Another problem is posed by the complexity of distributed systems of substation devices that exchange sensor information and autonomously decide on actuator controls. Certain malfunctions at this level may result in loss of equipment and the previously described outages.

There is an immediate need for research on the protection of critical infrastructure within the power grid to counter cyber-physical attacks and distributed control problems that may result in longer-lasting outages. We currently observe a complete absence in solutions to the problems discussed above. More so, no research focuses on these problems to date. One of the main causes is a lack of adequate simulation infrastructure to foster academic to contribute viable solutions. Hence, we recommend that a software simulation framework for the IEC 61850 standard be design at the level of substation devices, their interaction and their relation to and communication with control centers. This activity should be coordinated with a concerted effort by industry leaders providing valuable input on practicality and requirements. (The U.S. is lagging behind Europe where the CRISP project has filled this critical gap.) The resulting framework then needs to be complemented by initiatives to support follow-on research on possible cyber-attacks or distributed control problems within the power grid at the simulation level, the development of counter-measures at the software level and their integration into future standards as well as commercial deployment.

II. Verification and Validation of Worst-Case Execution Times

Current software design for safety-critical embedded systems requires stringent compliance with coding standards to ensure safety and reliability. One example is avionics where the RTCA DO-178B standard requires coverage testing (for statements, branches and conditionals). A very important additional requirement for real-time embedded systems is predictable timing behavior of software components. In particular, so-called hard real-time embedded systems have **timing constraints that must be met or the system is may malfunction**. Airbus (and likely also Boeing in the near future), *e.g.*, requires their sup-

pliers to provide verifiable bounds on worst-case execution time (WCET) for software to be deployed on planes currently under development (Airbus 380 and Boeing 787). The automotive industry is currently considering similar requirements, and others are likely to follow.

Determining bounds on the WCET of embedded software is a critically important problem for next-generation embedded real-time systems [1]. Currently, practitioners resort to testing methods to determine execution times of real-time tasks. However, testing alone cannot provide a verifiable (safe) upper bound on WCET. Exhaustive testing of inputs is generally infeasible, even for moderately complex input spaces due to its exponential complexity.

In contrast to dynamic testing, static timing analysis can provide *safe* upper bounds on the WCET of code sections, real-time tasks or entire applications. Hence, static timing analysis provides a safer and more efficient alternative to testing [2]. It yields verifiable bounds on the WCET of tasks regardless of program input by simulating execution along the control-flow paths within the program structure while considering architectural details, such as pipelining and caching [3].

These WCET bounds should also be *tight* to support high utilizations when determining if tasks can meet their deadlines *via* schedulability analysis. Tight bounds, however, can only be obtained if the behavior of hardware components is predicated accurately, yet conservatively with respect to its worst-case behavior. **Static timing analysis techniques are constantly trailing behind the innovation curve in hardware.** It is becoming increasingly difficult to provide tight *and* safe bounds in the presence of out-of-order execution, dynamic branch prediction and speculative execution. Simulation of hardware components is also prone to inaccuracy due to lack of information about subtle details of processors.

We advocate research on new approaches to bounding the WCET. Most importantly, a realistic hybrid approach is needed that combines formal static timing analysis with concrete micro-timing observations of actual architectures. First, a formal approach guarantees correctness. Second, dynamic timings on actual processors for small code sections will allow advanced embedded processor designs to be used in such time-critical systems, even in the presence of dynamic and unpredictable execution features. Third, any architectural modifications in support of such a paradigm have to be realistic in that they should reuse existing infrastructure both on the architecture side and the methodology for static timing analysis. There is an immediate need to develop software tools that can provide verifiable execution times to allow validation of task schedules within time-critical embedded systems. (The U.S. is lagging behind Europe in transfering research knowledge on WCET to products. However, the European results are also subject to trailing behind the hardware innovations curve, which underlines the need for research.)

III. Protection Against Soft Errors

Transient faults are becoming an increasing concern of system design for two reasons. First, smaller fabrication sizes have resulted in lower signal/noise ratio that more frequently leads to bit flips in CMOS circuits [4]. Second, embedded systems are increasingly deployed in harsh environments causing soft errors due to lack of protection on the hardware side [5]. The former reason affects computing at large while the latter is predominantly of concern for critical infrastructure. For example, the automotive industry has used temperature-hardened processors for control tasks around the engine block while space missions use radiation-hardened processors to avoid damage from solar radiation.

Current trends indicate an increasing rate of transient faults (*i.e.*, soft errors), not only due to smaller fabs but also because embedded systems are deployed in harsh environments they were not designed for. In commercial aviation, the next-generation planes (Airbus 380 and Boeing 787) will deploy off-the-shelf embedded processors without hardware protection against soft errors. Even though these planes are specifically designed to fly over the North Pole where radiation from space is more intensive due to a thinner atmosphere, target processors lack error detecting/correcting capabilities. Hence, system developers have been asked to consider the effect of single-event upsets (SEUs), *i.e.*, infrequent single bit flips, in their software design.

In practice, future systems may have to sustain transient faults due to any of the above causes. There exists a significant amount of work on detection of and protection against transient faults. Hardware can protect and even correct transient faults at the cost of redundant circuits [6–14]. Software approaches can also protect/correct these faults, *e.g.*, by instruction duplication or algorithmic design [15–21]. Recent work focuses at a hybrid solution of both hardware and software support to counter transient faults [22–24]. Such hybrid solutions aim at a reduced cost of protection, *i.e.*, cost in terms of extra die size, performance penalty and increased code size.

We advocate novel research to address the problem of soft errors. Of interested are (1) software solutions and (2) hybrid hardware/software solutions. While a number of hardware solutions exist, commodity hardware is being deployed in systems subject to high rates of transient errors. In the complete absence of hardware support, **a software methodology to address soft errors needs to be developed that retains performance.** Current software schemes (*e.g.*, [16]) reduce the performance of systems considerably, if not prohibitively, and are not supported by tools. Further research is required to reduce this overhead to developing novel schemes to tolerate faults in software. **Hybrid solutions offer another promising avenue to address this problem. Minor architectural modifications that can be adopted within existing architectures should be accompanied by software solutions allowing soft errors to be detected at low overhead.** Early results [22, 23] outline the potential of such an approach but leave many facets for improvement open. Protection at the level of code and different data sections of programs can be specialized by tool support to significantly reduce overhead even further. **There is an immediate need to pursue innovative lines of research for soft error protection that have potentially high yields in performance while providing low error rates.**

IV. Potential Impact

The outlined need for an open software simulation framework for the power grid addresses pressing economic and safety threats in one of our most important critical infrastructures. Solutions would contribute to our society and economy at large. The proposed directions of research on verifiable execution times would benefit the embedded system community, specifically applications in avionics, automotive and safety-critical systems. Solutions to the soft error problem will benefit the increasing set of embedded applications in harsh environments, which comprise critical infrastructure in today's society. This is particularly true for to aircraft using commodity microprocessors for control systems. The results can further benefit the semi-conductor industry at large by complementing their efforts to counter problems, such as the decreasing signal/noise ratio in smaller fabrication feature sizes, with innovative, cost-effective methods.

References

[1] J. Wegener and F. Mueller, "A comparison of static analysis and evolutionary testing for the verification of timing constraints," *Real-Time Systems*, vol. 21, no. 3, pp. 241–268, Nov. 2001.

[2] R. Arnold, F. Mueller, D. B. Whalley, and M. Harmon, "Bounding worst-case instruction cache performance," in *IEEE Real-Time Systems Symposium*, Dec. 1994, pp. 172–181.

[3] F. Mueller, "Timing analysis for instruction caches," *Real-Time Systems*, vol. 18, no. 2/3, pp. 209–239, May 2000.

[4] C. Constantinescu, "Trends and challenges in vlsi circuits reliability," *IEEE Micro*, pp. 14–19, July-August, 1996.

[5] V.Narayanan and Yuan Xie, "Reliability concerns in embedded system designs," *IEEE Computer magazine*, pp. 106–108, January, 2006.

[6] Hisashige Ando, Yuuji Yoshida, Aiichiro Inoue, Itsumi Sugiyama, Takeo Asakawa, Kuniki Morita, Toshiyuki Muta, Tsuyoshi Motokurumada, Seishi Okada, Hideo Yamashita, Yoshihiko Satsukawa, Akihiko Konmoto, Ryouichi Yamashita, and Hiroyuki Sugiyama, "A 1.3ghz fifth generation sparc64 microprocessor," in *Design Automation Conference*, New York, NY, USA, 2003, pp. 702–705, ACM Press.

[7] Y.C. Yeh, "Triple-triple redundant 777 primary flight computer," in *1996 IEEE Aerospace Applications Conference. Proceedings*, 1996, vol. 1, pp. 293–307.

[8] Y. C. (Bob) Yeh, "Design considerations in boeing 777 fly-by-wire computers," in *IEEE International High-Assurance Systems Engineering Symposium*, 1998, p. 64.

[9] Joel R. Sklaroff, "Redundancy management technique for space shuttle computers," *IBM Journal of Research and Development*, vol. 20, no. 1, pp. 20–28, 1976.

[10] Todd M. Austin, "DIVA: A reliable substrate for deep submicron microarchitecture design," in *International Symposium on Microarchitecture*, 1999, pp. 196–207.

[11] Mohamed Gomaa, Chad Scarbrough, T. N. Vijaykumar, and Irith Pomeranz, "Transient-fault recovery for chip multiprocessors," in *International Symposium on Computer Architecture*, San Diego, CA, May 2003, pp. 98–109, ACM SIGARCH / IEEE CS, Published as Proc. 30th Ann. Intl Symp. on Computer Architecture (30th ISCA 2003), FCRC'03 ACM Computer Architecture News, volume 31, number 2.

[12] A. Mahmood and E. J. McKluskey, "Concurrent error detection using watchdog processors - A survey," *IEEE Transactions on Computers*, vol. 37, no. 2, pp. 160–174, 1988.

[13] Joydeep Ray, James C. Hoe, and Babak Falsafi, "Dual use of superscalar datapath for transient-fault detection and recovery," in *International Symposium on Microarchitecture*, 2001, pp. 214–224.

[14] Steven K. Reinhardt and Shubhendu S. Mukherjee, "Transient fault detection via simultaneous multithreading," in *International Symposium on Computer Architecture*, 2000, pp. 25–36.

[15] P. Shirvani, N. Saxena, and E. McCluskey, "Software-implemented edac protection against seus," *IEEE Transactions on Reliability*, vol. 49, no. 1, pp. 273–284, 2000.

[16] N. Oh, P. Shirvani, and E. McCluskey, "Error detection by duplicated instructions in super-scalar processors," *IEEE Transactions on Reliability*, vol. 51, no. 1, pp. 63–75, 2002.

[17] Rajesh Venkatasubramanian, John P. Hayes, and Brian T. Murray, "Low-cost on-line fault detection using control flow assertions," in *International On-Line Testing Symposium*, 2003, pp. 137–143.

[18] Joakim Ohlsson and Marcus Rimén, "Implicit signature checking," in *International Symposium on Fault Tolerant Computing*, 1995, pp. 218–227.

[19] Guilin Chen, Mahmut T. Kandemir, and Mustafa Karaköy, "Memory space conscious loop iteration duplication for reliable execution," in *Static Analysis Symposium*, 2005, pp. 52–69.

[20] Sri Hari Krishna Narayanan, Seung Woo Son, Mahmut Kandemir, and Feihui Li, "Using loop invariants to fight soft errors in data caches," in *Asia and South Pacific Design Automation Conference*, Shanghai, China, January 18–21, 2005, pp. 1317–1320.

[21] Jie S. Hu, Feihui Li, Vijay Degalahal, Mahmut T. Kandemir, Narayanan Vijaykrishnan, and Mary Jane Irwin, "Compiler-directed instruction duplication for soft error detection," in *Design, Automation and Test in Europe*, 2005, pp. 1056–1057.

[22] George A. Reis, Jonathan Chang, Neil Vachharajani, Ram Rangan, and David I. August, "SWIFT: Software implemented fault tolerance," in *International Symposium on Code Generation and Optimization*, 2005, pp. 243–254.

[23] George A. Reis, Jonathan Chang, Neil Vachharajani, Ram Rangan, David I. August, and Shubhendu S. Mukherjee, "Design and evaluation of hybrid fault-detection systems," in *International Symposium on Computer Architecture*, 2005, pp. 148–159.

[24] Jun Yan and Wei Zhang, "Compiler-guided register reliability improvement against soft errors," in *International Conference on Embedded Software*, 2005, pp. 203–209.