FMitF: Track I: A Pathway for Combining Formal, Static, and Dynamic Analysis of Real-World Embedded Systems: Collaboration Plan

1 PI Qualifications

PI Groce has a long history with formal methods, including involvement in design and development of well-known model checkers, and application of model checkers at NASA/JPL on flight software for the Mars rovers, including application of and development contributions to, the SPIN and CBMC model checkers. Groce was the lead for design and development of automated testing tools and specification languages for the Mars Science Laboratory (Curiosity Rover) project. He received his PhD under the late Turing laureate Edmund M. Clarke, Jr., as one of the first PhD graduates focusing on model checking for software using SAT and SMT solvers. More recently, he has primarily focused on developing algorithms and tools for automated software testing; he is a core member of the DeepState and TSTL design and development teams. He brings to this project practical experience using verification and testing tools on critical real-world systems, and engineering such tools to be usable by engineers who are domain experts, not verification or testing experts. Groce is the lead developer for multiple GitHub-hosted testing and verification tools (including DeepState, the focus platform for this project), all with more than 75 GitHub "stars"; he reports 50-100 bugs detected using such tools to open source projects each year, and has been awarded significant bug bounties for security-critical compiler bugs detected using his work. Groce was recently chosen by the Bitcoin Core team to investigate the effectiveness of their fuzzing framework.

Co-PI Nghiem has an extensive background in control and autonomous systems, and application of formal methods in control systems. He has long experience working with timed automata and the Uppaal tool family. He developed, and was granted a U.S. patent for, methods for testing and verifying temporal logic specifications of hybrid systems—systems with both continuous and discrete behaviors. He brings to this project advanced knowledge of real-time embedded systems and practical experience applying verification methods and tools to them. He will also apply our methods and approaches to a multi-agent robotics system.

Co-PI Flikkema's current work includes research in energy-efficient embedded systems and networks, inference of the embedding environment, wireless sensor/actuator networks for monitoring and control of environmental and ecological systems, and cybersecurity with focus on IoT. Like Co-PI Nghiem, he ensures that developed approaches will be suitable for engineers whose primary focus is *building working systems*.

2 Collaboration Plan

The PIs are all faculty in the School of Informatics, Computing and Cyber Systems at Northern Arizona University. They all work in the same building, which also hosts the SEGA and DISCOVER labs (as opposed to the field installations which are located in various places, some near Northern Arizona University, others more remotely). PIs Flikkema and Groce have previously collaborated on research together, and all three PIs have toured each others' labs and reviewed code developed by the other PIs, including "guided tours" of the SEGA work by PI Flikkema and students. PIs Flikkema and Nghiem are both PIs on the DISCOVER project.

PIs will meet weekly for technical discussion and project status updates, depending on project status (and already do so, in discussions of the preliminary work) to coordinate their efforts, and ensure that tasks in the work plan are proceeding correctly, and to receive feedback on current status of tools. These meetings will alternate between meetings with just PIs, focused on higher level decision-making, and meetings including all involved students, focusing on status of individual work packages. In addition to weekly project meetings, there will be special out-of-band meetings to demonstrate significant new functionalities in tools, or to focus particularly on application of functionalities to specific SEGA or DISCOVER components.

The existing SEGA implementation code and fault(s) form a core concern that helps focus PI interactions, and enables easier communication of technical results between static and dynamic analysis experts and experts in the embedded systems domain. PIs will also set up a project repository, separate from DeepState, SEGA, or DISCOVER code repositories (already in existence) and use PR, tagged Issues, and other GitHub-supported collaborative software development best practices to ensure documentation of development, and team awareness of code status for components. GitHub Actions-based testing will be used to ensure all project members are aware of build or correctness problems with project code. Graduate students, undergraduate students, and PIs will also share a Project Slack Instance, linked to the code repo, to make discussion of issues arising during the project even easier, outside of scheduled meeting times.

PI Groce is well experienced in coordination of complex research efforts focused on open-source tools, as he coordinates most efforts for the DeepState project, which has 16 contributors, including a number of professional developers at Trail of Bits, and has managed releases of flagship tools such as Trail of Bits' echidna smart contract fuzzer (both tools with more than 500 GitHub stars and with dozens of forks; echidna has more than 20 contributors).

To summarize, collaboration will be coordinated through multiple overlapping methods, to ensure success:

- Weekly project meetings (alternating between PIs and PIs + students)
- Meetings of PIs with involved students
- Demonstration and SEGA or DISCOVER-focused meetings

- GitHub repositories for this project and SEGA, DISCOVER, and DeepState extensions
- GitHub PRs and Issues to coordinate development
- CI (GitHub actions) for code status awareness
- Project Slack linked to GitHub repo and CI, for effective communication

2.1 Evaluation of Formal Methods and Field Advances

The measures of success are discussed in context in the Project Description, but can be summarized as: for fuzzing, specification, and model checking advances, the work itself can be evaluated using traditional measures (coverage, bugs detected, curve of bug discovery, etc.); this aspect will also be measured in terms of research output in terms of released tools and their adoption and publications in the field.

For the field contributions, tasks T3 and T4 *are* the evaluation aspect of our project. The successful application of WP1 and WP2 tools to the case studies is essentially the driving factor in determining our success in the project. The measure of success is: (1) faults detected and corrected; (2) functionality proven correct using CBMC, symbolic execution engines, or SPIN; (3) coverage and other measures of generated tests; and (4) reported usability and value by engineers, particularly students. For T4.3, evaluation will be based on comparison of extracted time automata skeletons with independently developed full models.

2.2 Risks and Mitigations

Based on the project goals, we view the primary risk as loss of focus on a coherent application to the case study problems. This project is to be driven by embedded systems engineering needs, since we view advances that enable real-world usability and cost effectiveness as they key missing features in the application of formal methods to embedded systems engineering. One major mitigation is that we have proposed a more intense three-year effort, which is more sustainable and less likely to be impacted by personnel or case study project direction changes than a more diffuse four-year effort would be. The tight structure of frequent PC communications is also directed towards mitigating this risk.

The second primary risk addressed by the plan is that while many research efforts do not require production-quality software, we aim to primarily operate through enhancements to existing, production-quality, tools. We view the three-year plan as one mitigation in the sense that it allows for a more coordinated development effort, with less risk of personnel changes at the student level. Second, as noted, we will be applying a more rigorous development process than is frequently used in academic software systems. Because advances in formal methods application to real software are frequently rooted in heuristic solutions (since the general problem is usually infeasible, due to, e.g., Rice's theorem), correct and robust tool implementation work is essential to proper evaluation of advances.