SEL Embedded Debugger

An exception handler for embedded systems

Schweitzer Engineering Laboratories

Team SEL Embedded-Debugger

Subham Behera, Daniel Ochoa, Howard Potter

I. Introduction

Schweitzer Engineering Laboratories creates world class products that enhance the power grid in many ways such as preventing problems like blackouts and implementing cutting-edge solutions which add cybersecurity and automation. This makes the grid very efficient, however it is not impervious. To minimize downtime and quickly resolve issues, preventative measures must be implemented. This is where the embedded debugger comes into picture.

II. Background and Related Work

As SEL provides services and hardware to critical infrastructure, it is important to harden and secure these systems. However, every system is susceptible to problems. Due to the nature of these systems, it is critical to resolve and prevent any software challenge. Capturing and preserving system information when issues are detected is an approach to create a more resilient system. Creating an exception handler will provide a means to collect system state information for diagnosis and system restoration, ensuring minimal disruption to critical operations.

The system on a chip that SEL provides their customers is built around an Arm Cortex-A53 APU and Arm Cortex-R5F RPU. These types of processors have facilities for exception handling through the Arm Exception Model, which this project will leverage for exception handling. With this model, it will be possible to develop recovery techniques that can capture system data when the system runs into an exception. TCL, a common scripting language used on embedded systems, will be used to read the snapshot information collected on after a fault detection, for both debugging and to restore CPU core to its state.

By building upon these existing models that are already in use, a system that can take a snapshot of a system for restoration and debugging is achievable.

As this type of project is concerned with a lower level of programming, our team will have to learn and improve our skills when it comes to scripting languages. While our team does have experience in working with machine language, we are not familiar with the ARM architecture, which will be critical in developing this project. Additionally, TCL will be a new language for this team – we are familiar with other scripting languages, but it will still provide a challenge. There is also the challenge of debugging hardware on this level, a methodology we are not familiar with. We are certain these goals are achievable, and that a usable debugging system that can be used by SEL's system operations.

III. Project Overview

SEL develops industrial-grade devices designed to protect critical infrastructure. These devices, like many embedded systems, occasionally encounter software faults that need to be resolved efficiently. To improve SEL's ability to diagnose and fix these issues, we want to implement an exception handler for their embedded platform. This handler should capture and store valuable system information related to the CPU and memory when a software issue is detected.

Our exception handler will trigger when software issues arise and capture vital system information from the processor. The main objective is to create a thorough system snapshot that retains system data, including essential information about CPU core registers and associated memory. The exception handler will also include scripts that can read the snapshot we captured

and use TCL commands and a debugger to restore the CPU core to the exact state captured in the snapshot. This is to ensure minimal downtime when a system runs into a fault, as well as to collect this data to diagnose the system fault.
The project will include a well-documented test setup that confirms the functionality of the entire system, ensuring that the captured information can be effectively employed in debugging and issue resolution. These outcomes collectively serve to fortify SEL's ability to provide efficient solutions for software challenges in critical infrastructure protection devices while also providing valuable embedded software development and debugging experience to the team.

IV. Client and Stakeholder Identification and Preferences

The primary client is Schweitzer Engineering Laboratories. As stated above, they prefer the main code to be in C/C++ and it should use ARM assembly to communicate with the hardware. It is given that this project is a requirement, as having a method to restore systems and diagnose faults quickly will improve their product.

Other indirect stakeholders include consumers who use power that is supplied from power grids operated by Schweitzer Engineering Laboratories. With more advanced technology comes more responsibility and if the embedded debugger wouldn’t have been considered to develop, newer products would be completely wasted if they were to fail. This means more chances of blackouts and unreliable energy, affecting consumers.

V. Glossary

TCL – A high-level, dynamic programming language. It is commonly used in embedded systems for scripting, prototyping, testing, and debugging.

System On a Chip (SOC) – An integrated circuit that incorporates different components onto one board, as an integrated system.

VI. References

[1] “We Do Our Part So They Can Do Theirs, and Together We Power the Future,” *selinc.com*. https://selinc.com/company/our-part/
[2] “Arm Architecture Reference Manual for A-profile architecture,” Arm Developer, April 2023, available at https://developer.arm.com/documentation/ddi0487/latest/
[3] “Zynq UltraScale+ MPSoC Data Sheet”, AMD, November 2022, available at https://docs.xilinx.com/v/u/en-US/ds891-zynq-ultrascale-plus-overview