Chapter 7
HARDWARE PATCHING WITH FIELD-REPAIRABLE CONTROL LOGIC

Abstract. This chapter describes in detail field-repairable control logic (FRCL), a solution that we recently developed [WBA06]. FRCL is a patching-based runtime verification technique that relies on an on-die programmable matcher. The matcher compares the state of the processor to patterns describing known bugs. In particular, FRCL targets control bugs in microprocessor cores, which, as our analysis in Section 6.1 shows, dominate the landscape of escaped errors in commercial products. To detect such control bugs, the matcher is engineered to monitor multiple critical signals in a core’s control logic block. The patterns stored in the matcher are developed by the manufacturer after an escaped error is detected and diagnosed; patterns are then distributed to end-users via patches, such as BIOS updates. When a buggy situation is detected, the matcher recovers from it with a pipeline flush and invokes a degraded mode of operation. In this mode, the complexity of the processor is greatly reduced, sacrificing some performance features, but allowing to formally prove the complete functional correctness of the system. Once the buggy situation is bypassed, the processor resumes normal high-performance operation. We analyze different aspects of FRCL operation and describe a methodology for automatic selection of signals to be monitored by the matcher. Finally, we extend the field-repairable control logic framework with semantic guardians - hardware circuits encoding all control states of the design that have been verified prior to its release. With the help of the guardians the processor can be guaranteed to always operate in a verified state (in either normal or degraded mode), thus enabling trusted computation.

7.1 Introduction

Once a system is verified to a satisfactory level during its development, a design house will ramp up its production and start distributing it to customers. Unfortunately, due to shrinking development timelines and rapidly increasing complexity of modern processors, released designs are never fully and exhaustively verified and often contain subtle errors. These escaped bugs are eventually identified when
the system is already deployed in the field and cannot be easily corrected by the manufacturer, since they require modification of the actual silicon die. Recognizing the inevitability of such errors and the need to efficiently fix them without a costly product recall, researchers in recent years started to develop hardware patching solutions. Our analysis in Section 6.1 demonstrates that the majority of such escaped errors occur in the control logic portion of the design, which traditionally has been the hardest to verify. To enable efficient in-the-field patching of a processor’s control logic, this chapter presents a low-cost and expressive mechanism, called field-repairable control logic (FRCL), originally presented in [WBA06] and later extended in [WBA08]. In this framework, when an escaped bug is exposed in the field, the support team investigates it and generates a pattern describing the control state of the processor that triggers the occurrence of the bug. The pattern is then sent to end customers as a patch and is loaded into an on-die state matcher at system startup. The matcher constantly monitors the state of the processor and compares it to the stored patterns to identify when the pipeline has entered a state associated with a bug. Once the matcher has determined that the processor is in a flawed control state, the pipeline is flushed and forced into a degraded mode of operation for the execution of the subsequent instruction.

In degraded mode, the processor starts execution from the first uncommitted instruction and allows only one operation to traverse the pipeline at a time. Therefore, much of the control logic that handles interactions between operations can be turned off. The resulting system is greatly simplified, enabling a complete formal verification of the degraded mode at design time. In other words, we can guarantee that instructions running in this mode complete properly, and thus can ensure forward progress, even in the presence of design errors, by simply forcing the pipeline to run in degraded mode. After the error is bypassed in degraded mode, the processor returns to high-performance mode until the matcher finds another flawed control state. In designing the state matcher, we have devoted special attention to creating a system that can detect multiple design errors with minimal false positive triggering. In addition, for situations where the number of patterns of design errors exceeds the capacity of a given matcher, we developed a novel compression algorithm that compacts the erroneous state patterns while minimizing the number of false positives introduced by this process.

In Section 7.4 we extend the ideas of the field-repairable control logic technology further, presenting a solution that provides protection even from unknown escaped bugs in deployed systems. Instead of requiring a vendor company to identify an escaped bug before we can repair it, we simply assume now that any configuration that was not verified at design time is potentially a buggy configuration. Thus, in our semantic guardian solution, we protect a system against all those configurations that were not verified at design time. We will show in Section 7.5.6 that we can indeed deliver such protection at minimal performance costs, since, even if the unverified portion of a design is extensive, unverified configurations tend to occur rarely at run-time. Moreover, we discuss how we can match a large set of unverified configuration with little area overhead.