Handbook of FPGA Design Security

Ted Huffmire • Cynthia Irvine • Thuy D. Nguyen •
Timothy Levin • Ryan Kastner • Timothy Sherwood

Handbook of FPGA Design Security

Springer

Dr. Ted Huffmire
Department of Computer Science
Naval Postgraduate School
Cunningham Road 1411
93943 Monterey, CA
USA
tdhuffmi@nps.edu

Dr. Cynthia Irvine
Department of Computer Science
Naval Postgraduate School
Cunningham Road 1411
93943 Monterey, CA
USA

Thuy D. Nguyen
Department of Computer Science
Naval Postgraduate School
Cunningham Road 1411
93943 Monterey, CA
USA

Timothy Levin
Department of Computer Science
Naval Postgraduate School
Cunningham Road 1411
93943 Monterey, CA
USA

Dr. Ryan Kastner
Dept. of Computer Science and Eng.
University of California, San Diego
Gilman Drive 9500
92093 La Jolla, CA
USA
kastner@cs.ucsd.edu

Dr. Timothy Sherwood
Department of Computer Science
UC, Santa Barbara
93106 Santa Barbara
USA

ISBN 978-90-481-9156-7 e-ISBN 978-90-481-9157-4
DOI 10.1007/978-90-481-9157-4
Springer Dordrecht Heidelberg London New York

Library of Congress Control Number: 2010930170

Cover design: eStudio Calamar

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

To our teachers

Preface

The purpose of this book is to provide a practical approach to managing security in FPGA designs for researchers and practitioners in the electronic design automation (EDA) and FPGA communities, including corporations, industrial and government research labs, and academics. This book combines theoretical underpinnings with a practical design approach and worked examples for combating real world threats. To address the spectrum of lifecycle and operational threats against FPGA systems, a holistic view of FPGA security is presented, from formal top level specification to low level policy enforcement mechanisms, which integrates recent advances in the fields of computer security theory, languages, compilers, and hardware. The net effect is a diverse set of static and runtime techniques that, working in cooperation, facilitate the composition of robust, dependable, and trustworthy systems using commodity components.

We wish to acknowledge the many people who helped us ensure the success of our work on reconfigurable hardware security. In particular, we wish to thank Andrei Paun and Jason Smith of Louisiana Tech University for providing us with a Linux-compatible version of Grail+. We also wish to thank those who gave us comments on drafts of this book, including Marco Platzner of the University of Paderborn, and Ali Irturk and Jason Oberg of the University of California, San Diego. This research was funded in part by National Science Foundation Grant CNS-0524771 and NSF Career Grant CCF-0448654.

Monterey, CA, USA — Ted Huffmire
Cynthia Irvine
Thuy D. Nguyen
Timothy Levin
La Jolla, CA, USA — Ryan Kastner
Santa Barbara, CA, USA — Timothy Sherwood

Ted Huffmire is an assistant professor of computer science at the Naval Postgraduate School in Monterey, California. His research spans both computer security and computer architecture, focusing on hardware-oriented security and the development of policy enforcement mechanisms for application-specific devices. He has a Ph.D. in computer science from the University of California, Santa Barbara. He is a member of the IEEE and the ACM.

Cynthia Irvine is the director of the Center for Information Systems Security Studies and Research (CISR) and a professor of computer science at the Naval Postgraduate School in Monterey, California. Her research interests include high-assurance security. She has a Ph.D. in astronomy from Case Western Reserve University. She is a member of the IEEE, the ACM, and the Astronomical Society of the Pacific.

Thuy D. Nguyen is a senior researcher of computer science at the Naval Postgraduate School in Monterey, California. Her research interests include high-assurance platforms, trusted operating systems, dynamic security services, multilevel security, security evaluation, and security requirements engineering. She has a B.A. in computer science from the University of California, San Diego.

Timothy Levin is an associate research professor at the Naval Postgraduate School in Monterey, California. His research interests include design, analysis and verification of high-assurance security architectures and dynamic security policies. He has a B.S. in computer science from the University of California, Santa Cruz. He is a member of the IEEE and the ACM.

Ryan Kastner is an associate professor in the Department of Computer Science and Engineering at the University of California, San Diego. His research interests focus on many aspects of embedded computing systems, including reconfigurable architectures, digital-signal processing, and security. He has a Ph.D. in computer science from the University of California, Los Angeles.

Timothy Sherwood is an associate professor in the Department of Computer Science at the University of California, Santa Barbara. His research interests include computer architecture, specifically in the development of novel high-throughput methods by which systems can be constructed, monitored, and analyzed. He has a Ph.D. in computer science and engineering from the University of California, San Diego. He is a member of the IEEE and the ACM.

Contents

Acronyms

ACL	Access Control List.
ACM	Association for Computing Machinery.
AES	Advanced Encryption Standard. A common symmetric crypto algorithm.
API	Application Programming Interface.
ASIC	Application-Specific Integrated Circuit. An integrated circuit with "hard-wired" functionality intended for use in one or a limited number of applications.
BBV	Basic Block Vector.
B&L	Bell-LaPadula confidentiality model.
BRAM	Block Random Access Memory.
C&A	Certification and Accreditation.
CAD	Computer-Aided Design.
CAP	Capability Protection.
CC	Common Criteria.
CCEVS	Common Criteria Evaluation and Validation Scheme.
CIA	Confidentiality-Integrity-Availability triad.
CLB	Configuration Logic Block. The basic repeated building block of an FPGA, usually consisting of LUTs, registers, and other logic.
CM	Configuration Management.
CMP	Chip Multiprocessor.
COI	Conflict-of-Interest Class.
COTS	Commercial Off-the-Shelf.
CPU	Central Processing Unit. A general-purpose processor.
DARPA	Defense Advanced Research Projects Agency.
DFA	Deterministic Finite Automaton.
DoS	Denial-of-Service.
DRAM	Dynamic Random Access Memory.
DSP	Digital Signal Processing.
DVI	Digital Visual Interface.

EAL	Evaluation Assurance Level.
EDK	Embedded Development Kit.
EPROM	Erasable Programmable Read-Only Memory.
EEPROM	Electrically Erasable Programmable Read-Only Memory.
ESL	Electronic System Level. ESL Design involves the compilation of a high-level specification of a system to a low-level hardware implementation.
FEMA	Federal Emergency Management Agency.
FFT	Fast Fourier Transform.
FIPS	Federal Information Processing Standard.
FPGA	Field Programmable Gate Array. A reconfigurable hardware device created as an array of logic blocks tied together with a programmable interconnect.
FSA	Finite State Automaton.
FTLS	Formal Top Level Specification. A description of a policy that specifies the legal sharing of memory among cores on an FPGA.
HDL	Hardware Description Language. A language for designing hardware components.
I&A	Identification and Authentication.
IBM	International Business Machines.
IC	Integrated Circuit.
ICAP	Internal Configuration Access Port. An interface to an FPGA for dynamic partial reconfiguration.
IDS	Intrusion Detection System.
IEEE	Institute of Electrical and Electronics Engineers.
I/O	Input/Output.
IOB	Input/Output Block.
IP	Intellectual Property. Hardware modules that are the building blocks of an embedded design.
ISA	Instruction Set Architecture.
ISE	Integrated Synthesis Environment.
IT	Information Technology.
LCD	Liquid Crystal Display.
LED	Light-Emitting Diode.
LFU	Least Frequently Used.
LPSK	Least Privilege Separation Kernel.
LRU	Least Recently Used.
LUT	Look-Up Table. The smallest FPGA logic component, the LUT can be programmed to imitate any logic gate by directly storing the truth table for that gate.
LVS	Layout-versus-schematic. Comparison tools recommended by Trimberger for detecting subversion in FPGA designs via non-destructive validation of the equivalence between the original design and the implemented design.

MATLAB Matrix Laboratory. Mathematical software that can be used for developing DSP algorithms.
MIMO Multiple Input Multiple Output.
MLS Multilevel Security. The science of building systems that can process data with different security labels (e.g., SECRET and UNCLASSIFIED).
NFA Nondeterministic Finite Automaton.
NIAP National Information Assurance Partnership.
NIST National Institute of Standards and Technology.
NP Nondeterministic Polynomial.
NRE Non-Recurring Engineering. The masks used in ASIC fabrication are a large part of the NRE cost of ASICs.
NSA National Security Agency.
NSTISSP National Security Telecommunications and Information Systems Security Policy.
OPB On-chip Peripheral Bus.
PCI Peripheral Component Interconnect.
PKI Public Key Infrastructure.
PUF Physical Unclonable Function. A unique number generated from variations in the manufacturing process.
RAM Random Access Memory.
RISC Reduced Instruction Set Computer.
RVM Reference Validation Mechanism.
RSA Rivest Shamir Adelman. A common asymmetric key crypto algorithm.
SDK Software Development Kit.
SGI Silicon Graphics, Inc.
SKPP Separation Kernel Protection Profile.
SMP Symmetric Multiprocessor.
SMT Simultaneous Multithreading.
SoC System-on-a-Chip.
SRAM Static Random Access Memory.
SRC Seymour Roger Cray. Founder of SRC Computers.
SSL Secure Sockets Layer.
TCB Trusted Computing Base.
TCSEC Trusted Computer System Evaluation Criteria.
TIC TRUST in Integrated Circuits. A DARPA program concerned with hardware subversion.
TLB Translation Lookaside Buffer.
TOE Target of Evaluation.
TSF TOE Security Functionality. A set consisting of all hardware, software, and firmware of the TOE that must be relied upon for the correct enforcement of the security functional requirements.
USB Universal Serial Bus.
VHDL VHSIC (Very-High-Speed Integrated Circuits) Hardware Description Language.
VLIW Very Long Instruction Word.

VM	Virtual Machine.
VMM	Virtual Machine Monitor.
VPN	Virtual Private Network.
VPR	Versatile Place and Route.
WAP	Wireless Access Point.
XPS	Xilinx Platform Studio.

Chapter 1
Introduction and Motivation

Abstract From Bluetooth transceivers to the NASA Mars Rover, FPGAs have become one of the mainstays of embedded system design. By merging properties of hardware and software, reconfigurable devices provide an attractive tradeoff between the performance of application-specific hardware and the programmability of CPUs. Although this flexibility allows developers to quickly prototype and deploy embedded systems with performance close to ASICs, this programmability can also be exploited to disrupt critical functionality, eavesdrop on encrypted communication, or even destroy a chip. Creating systems which are both efficient and flexible, yet fundamentally sound from a security point of view, is an exceedingly challenging endeavor for both researchers and practitioners. All too often the security aspects of a reconfigurable design are not addressed until far too late in the design process, resulting in systems that are protected only by their obscurity. This chapter presents an overview of Field Programmable Gate Array (FPGA) technologies from the viewpoint of security, specifically how and why these devices have grown in importance over the last decade to become one of the most trusted and critical elements of modern computer systems. This chapter also discusses their changing role from a platform for prototyping to a deployable solution, the architecture of a modern FPGA, the security ramifications of their increased use, and some of the lessons from the security community that may be applicable in this domain.

1.1 The Growing Reliance on FPGAs

FPGAs are at the heart of many mission critical devices, silently controlling everything from wireless access points (WAP) to commercial face recognition systems. Unlike the sequential execution provided by a general purpose processor, modern Field Programmable Gate Arrays can perform hundreds of multiplies and thousands of adds each cycle, giving them the computational power to host many different logic modules at the same time. For example, an FPGA-hosted Wireless Access Point (WAP) may employ a signal processing core, a protocol pro-

T. Huffmire et al., *Handbook of FPGA Design Security*,
DOI 10.1007/978-90-481-9157-4_1,

cessing engine, and a packet scheduler, all sharing the same physical silicon. Furthermore, because reconfigurable hardware can be rewritten in both the lab and in the field, rapid design cycles are possible, and patches can even be downloaded to devices already deployed (e.g. bug fixes or functionality enhancements can be pushed out over the network to cell phones or wireless access points on demand).

Because of this rare combination of computation power and flexibility, reconfigurable devices are now the workhorses behind a broad variety of performance-critical embedded systems [9, 15, 19, 40, 51, 62]. In fact, many reconfigurable machines achieve 100x speedups and 100x performance gain per unit of area as compared to a similar microprocessor [12, 18, 75]. Satellites, set-top boxes, intrusion detection systems, the electrical power grid, cryptography units, aircraft, and even the Mars Rover all rely on Field Programmable Gate Arrays (FPGAs) to perform their respective functions. It is estimated that in 2005 alone there were over 80,000 different commercial FPGA design projects started [53]. The bit-level reconfigurability of these devices can be used to implement highly optimized circuits for everything from encryption to FFTs, or even entire customized multi-processor systems. In this section we describe a couple of these different domains and how FPGAs are used in them.

> *Design Tip*: *Benefits of FPGAs*. FPGAs are ideal for rapid protyping of embedded designs and the development of novel computer architectures. The increasing cost of ASIC manufacturing, the performance advantages of FPGAs over general-purpose processors, and the narrowing performance gap between FPGAs and ASICs have resulted in the growing use of FPGAs in real systems. For low-volume segments of the market, such as highly trustworthy systems, FPGAs provide both cost and security advantages over ASICs. For example, with FPGAs, sensitive designs are never sent to a foundry where they could be stolen. The parallelism available on FPGAs also makes them an attractive alternative to general-purpose CPUs for throughput-driven applications.

1.1.1 FPGAs for Aerospace

Because FPGAs are able to provide a useful balance between performance, cost, and flexibility, many avionics systems now make use of them. For example, FPGAs perform crucial functions in the Joint Strike Fighter [56], the new Boeing 787 Dreamliner [20], and the NASA Mars Rover [23, 57]. In these applications, FPGAs are used for cockpit displays, flight management, avionics, weapons guidance, and flight radar [2].

Design Tip: *FPGA Basics*. A *bitstream* specifies how to set the configuration bits of the FPGA fabric. A *core* is a circuit that is used as a building block of a larger embedded system. An *antifuse-based* FPGA uses fuses as the configuration bits; therefore, it can only be programmed once and is nonvolatile thereafter. An *SRAM-based* FPGA uses volatile SRAM cells as the configuration bits. A *flash-based* FPGA uses EEPROM cells as the configuration bits.

The circuits may be either antifuse-based, flash-based, or SRAM-based. While fuse-based circuits are write-once devices, SRAM- and flash-based FPGAs can be written many times, either in the lab or in the field. Flash-based circuits provide low power advantages.

Design Tip: *Choosing an FPGA Type*. SRAM-based, flash-based, and antifuse-based FPGAs have different security properties [76]. Despite the limitation that it is a write-once technology, an antifuse FPGA offers the advantage that theft of the design requires a painstaking and destructive *sand-and-scan* attack, involving removal of the packaging and the progressive etching and electron micrography of each layer to create a 3-D image of the chip. Since the fuses are nonvolatile, the bitstream does not have to be loaded from off-chip, exposing it to board-level probing attacks and attacks on the bitstream encryption mechanisms. Flash-based FPGAs also can store the bitstream on-chip, and the design does not have to be loaded from off-chip. This eliminates one avenue for attackers. However, unlike an antifuse-based FPGA, the design is changeable since flash memory can be modified. In addition, flash-based FPGAs are much easier to probe, and probing attacks are much cheaper to carry out than sand-and-scan attacks. SRAM-based FPGAs must load the bitstream every time they are powered on, and soft memory errors [28] or flaws in the implementation of bitstream decryption mechanisms may provide an opportunity for a well-funded adversary to extract the design. When powered continuously, SRAM-based FPGAs are similar to non-volatile FPGAs in that the design does not have to be loaded from non-volatile off-chip memory.

Consider the example of military avionics, in which a single chip processes both classified targeting information and unclassified fueling and maintenance information. Other multilevel security (MLS) scenarios for avionics include the sensor-shooter problem, in which the intelligence analysts who decide on targets have higher clearances than the soldiers ordered to attack those targets. In another MLS scenario, a coalition member flies in formation with his or her allies, and a policy

must specify what information may be shared with each of them. In these multi-level systems, a CPU can be *allocated* to handle a particular level (or a range of levels) of data, and it is assigned a security label (or a security label range). A separate device for each security level adds too much weight to an aircraft. To minimize weight, a single device that can process data at multiple levels is attractive, but without careful attention to security, can be dangerous. Keeping the different levels of information separate requires careful design. Since reconfigurable systems often lack the memory protection, virtual memory, and other traditional separation mechanisms available in a general-purpose system, security techniques are needed to prevent classified data from mixing with unclassified data. In addition, like software update mechanisms, it is critical that the process for remotely updating these devices is very secure to prevent sabotage. Finally, due to the sensitive nature of the intellectual property, it is very important to keep competitors or enemies from being able to reverse engineer these systems easily.

1.1.2 FPGAs for Supercomputing

While desktop computers continue to make incredible gains in performance, there are always problems that lie outside the capabilities of these machines, and scientists and engineers eventually turn to "big iron" when performance is needed. Many supercomputer companies, including SRC Computers [25, 71], Cray [58], and SGI [67, 68], have integrated reconfigurable hardware into their systems to improve performance [10, 16]. A good example of such a system is Cray's XD1 architecture, which combines six large Xilinx FPGAs (Virtex-4) with twelve x86 processors in each chassis. When an application is loaded on the machine, it includes a bitstream for programming the associated reconfigurable hardware. Although the past generations of FPGAs were not cost competitive with microprocessors in delivering double precision floating point [74], they can provide significant improvements (100x) in integer dominated applications. The current generation of FPGAs includes more integrated support for floating point. In the supercomputing environment, often the code and data being run are either sensitive intellectual property or even classified in nature, requiring a commensurately secure computing environment. Furthermore, supercomputing centers require strong physical security because they are very high profile targets for intruders.

The SRC Reconfigurable computer is an example of a system that uses FPGAs to provide acceleration for programs running on general-purpose processors [25, 71]. Logging into the machine is via a traditional Unix shell interface. A project folder contains both Verilog and either C or Fortran code. A Makefile invokes a Verilog compiler for the Verilog code (resulting in a bitstream), and it also invokes a C compiler for the C code (or a Fortran compiler for the Fortran code). Executing a program on the SRC requires loading the bitstream onto the reconfigurable hardware and loading the executable program onto the general-purpose hardware. From a security standpoint, if the host OS, application software, or user account has been