Preventing Piracy and Reverse Engineering of SRAM FPGAs Bitstream

Lilian Bossuet¹, Guy Gogniat¹, Wayne Burleson²

¹ LESTER Lab
Université de Bretagne Sud
Lorient, France
lilian.bossuet@univ-ubs.fr

² Department of Electrical and Computer Engineering
University of Massachusetts, Amherst, USA
burleson@ecs.umass.edu
Outline

◆ Design Security
◆ Actual Solutions
◆ Propositions
◆ Conclusion
FPGA and Security ...

Use the FPGA to protect
- Network isolation (firewalls)
- Smart cards
- Sensor networks

Protect the FPGA
- Data encryption scheme
- Protection against hardware damage

Protect the FPGA configuration
- Bitstream encryption
Design Security
Need for Design Security

Protection against

- **Cloning**
  - A competitor makes copy of the boot ROM or intercepts the bitstream and copies the code.

- **Reverse Engineering**
  - A competitor copies a design by reconstructing a “schematic” or netlist level representation; in the process, he understands how the design works and how to improve it, or modify it with malicious intent.

Attack Types

- **Noninvasive**
  - Monitored by external means such as brute force key generation, changing voltages to discover hidden test modes, etc.

- **Invasive**
  - Decapped and then microprobed using focused ion beams, or other sophisticated techniques to determine the contents of the device.
Levels of Semiconductor Security

- #3 Can be broken into in a government sponsored lab (e.g. USA NSA or France DGA)
- #2 Can theoretically be broken into with time and expensive equipment
- #1 Not secure, easily compromised with low costs tools

Source: IBM systems journal Vol. 30 No 2 - 1991
# Actual Integrated Circuits Level of Security

*Source: ACTEL*

<table>
<thead>
<tr>
<th>DEVICES</th>
<th>SECURITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRAM FPGA</td>
<td>LEVEL 1</td>
</tr>
<tr>
<td>ASIC Gate Array</td>
<td>LEVEL 2 -</td>
</tr>
<tr>
<td>Cell-Based ASIC</td>
<td>LEVEL 2 -</td>
</tr>
<tr>
<td>SRAM FPGA with Bitstream encryption</td>
<td>LEVEL 2</td>
</tr>
<tr>
<td>Flash FPGA</td>
<td>LEVEL 2 +</td>
</tr>
<tr>
<td>Antifuse FPGA</td>
<td>LEVEL 2 +</td>
</tr>
</tbody>
</table>
Antifuses FPGA

- Very good for design security
  - No bitstream can be intercepted in the field (no bitstream transfer, no external configuration device)
  - Need a Scanning Electron Microscope (SEM) to try to know the antifuse states (an Actel AX2000 antifuse FPGA contains 53 million antifuses with only 2\(^5\) \% programmed in an average design)

- BUT not really “reconfigurable” just configurable ...

FPGA MARKET SHARE (2000)

- Altera: 34\%
- Xilinx: 38\%
- Lattice: 14\%
- Other: 8\%
- Actel: 6\%
- One Time Programmable FPGA: \(\approx 10\%\)
- SRAM FPGA: \(\approx 90\%\)
Actual Solutions
Xilinx Solution

- **Need of an external battery to save the keys**
- **The decryption circuit takes FPGA resources (silicon)...**
- **No flexibility for the decryption algorithm**
- **Partial reconfiguration is no more available**

Protection against cloning and reverse engineering
Altera Solution

- The decryption circuit takes FPGA resources (silicon)...
- No flexibility for the decryption algorithm
Algotronix (U.K.) Proposition

- A secret cryptographic key can be stored on each FPGA
- No need for the CAD tools or any person to have knowledge of the key
- The Encryption Decryption circuit takes FPGA resources (silicon)...
- No flexibility for the decryption algorithm

Propositions...
Desirable Characteristics

- Strong protection against cloning and reverse engineering for SRAM FPGA
- No additional battery => the keys are embedded (laser)
- Choice of the suitable encryption algorithm and architecture (security policy)
- Keep free the FPGA application-dedicated resources
Application Security Policy

Actual solutions: the whole design (all IPs) has the same security policy

Proposition: Each IP can be encrypted with its own algorithm
Security Policy

- Application partitioning with necessary security levels
- All of the application doesn’t need to be encrypted (free available IP)

Security-Critical Part:
your Intellectual property which needs higher security due to
- development cost
- potential security breach

No critical Parts:
Generic functions and cores such as communication protocols, etc.
Encryption - Decryption management

- Once the FPGA is configured encryption and decryption circuits do not use FPGA resources
- Use of partial dynamic reconfiguration
- Use of self-reconfiguration (to configure each IP with the corresponding decryption circuit)
- Embedded secret key
Encryption Management (in the lab.)

- **IP1** (SCP1) algorithm 1
- **IP2** (SCP2) algorithm 2
- **IP3** (NCP)

![Encryption Circuit 1](image)

FPGA: secret KEY

**EPROM**
- SCP1 Encrypted
- Decryption Circuit 1
Encryption Management (in the lab.)

- IP1 (SCP1) algorithm 1
- IP2 (SCP2) algorithm 2
- IP3 (NCP)

Encryption circuit2

FPGA: secret KEY

EPROM
- SCP1 Encrypted
- Decryption circuit1
- SCP2 Encrypted
- Decryption circuit2

http://lester.univ-ubs.fr/
Encryption Management (in the lab.)

- **IP1** (SCP1) algorithm 1
- **IP2** (SCP2) algorithm 2
- **IP3** (NCP)

**EPROM**
- SCP1 Encrypted
- Decryption circuit1
- SCP2 Encrypted
- Decryption circuit2
- NCP

FPGA: secret KEY
Decryption Management (at power up)

- **EPROM**
  - SCP1: Encrypted
  - Decryption circuit1
  - SCP2: Encrypted
  - Decryption circuit2
- **NCP**
- **Configuration Controller**
  - FPGA: secret KEY
  - Decryption circuit1
Decryption Management (at power up)

- Partial and self reconfiguration

EPROM

SCP1
Encrypted

Decryption circuit1

SCP2
Encrypted

Decryption circuit2

NCP

Configuration Controller

IP1

FPGA: secret KEY

Decryption circuit1
Decryption Management (at power up)

- Partial and self reconfiguration
- The decryption circuit1 is replaced by the decryption circuit2

Diagram:
- EPROM
  - SCP1 Encrypted
  - Decryption circuit1
  - SCP2 Encrypted
  - Decryption circuit2
- Configuration Controller
- IP1
  - NCP
  - FPGA: secret KEY
  - Decryption circuit2
Decryption Management (at power up)

- Partial and self reconfiguration
- The decryption circuit1 is replaced by the decryption circuit2

**EPROM**

- SCP1 Encrypted
- Decryption circuit1
-SCP2 Encrypted
- Decryption circuit2

**Configuration Controller**

- IP1
- FPGA: secret KEY
- Decryption circuit2

**IP2**

- NCP
Decryption Management (at power up)

- Partial and self reconfiguration
- The decryption circuit1 is replaced by the decryption circuit2
- The decryption circuit2 is replaced by the non critical part that is not encrypted

- EPROM
- SCP1 Encrypted
- Decryption circuit1
- SCP2 Encrypted
- Decryption circuit2
- NCP
- Configuration Controller
- IP1
- IP2
- IP3
- FPGA: secret KEY
Main issues

- Partial and self-reconfiguration
- Configuration controller to perform the configuration management
- Area related to the decryption algorithms
- Keys management for each encryption/decryption circuit
Area related to the decryption algorithms

- Once the last encrypted bitstream is configured the last decryption algorithm can be replaced by non-critical parts.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>#slices of the cryptographic core</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rijndael</td>
<td>4312 5302 2902</td>
</tr>
<tr>
<td>Serpent</td>
<td>1250 7964 4438</td>
</tr>
<tr>
<td>RC6</td>
<td>1749 3189 1139</td>
</tr>
<tr>
<td>Twofish</td>
<td>2809 3053 1076</td>
</tr>
</tbody>
</table>

~ 100 Mbits/s < Throughput < ~400 Mbits/s

- ICAP performance: 66 Mbits/s, throughput is not the major concern but area
Keys management and protection

- Secret key is defined during the fabrication process (laser)
- Define a large key e.g. 1024 bits that is used to generate each secret key (56 bits, 128 bits)
- Hardwired key generator
- Hardwired mechanism to authenticate each decryption circuit and encrypted bitstream (signature)
Advantages / Inconveniences

- The encryption/decryption circuit doesn’t take FPGA resources
- Choice of a suitable encryption algorithm and architecture (to obtain a required security level)
- Only designer knows the chosen algorithm and architecture
- The system can be upgraded with new encryption algorithms
- No additional battery with an embedded secret key

- Complex system, keys management
- Management of partial, self and dynamic reconfiguration
- Take time during the system set up
Conclusion...
Conclusion

- In consequence of the rise of the FPGA, it is necessary to prevent piracy and reverse engineering of FPGA Bistream

- Existing SRAM FPGA are insecure

- We propose original solutions that take advantage of the partial and dynamic FPGA configuration, with a no-fixed encryption algorithm and architecture

- It is a first step toward security policy for FPGA and the solutions still to be improved ...
Comments?

Questions?

Thank you