Beyond the Focus Penetration Testing in Future Hardware
Fuzzing the RTL

Mary Yeoh
Intel Penang Design Center (iPDC)
Intel Corporation
Penang, Malaysia
Legal Disclaimer
Today's presentation may contain forward-looking statements. All statements made that are not historical facts are subject to a number of risks and uncertainties, and actual results may differ materially. Please refer to our most recent Earnings Release and our most recent Form 10-Q or 10-K filing available on our website for more information on the risk factors that could cause actual results to differ.

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL’S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Intel products are not intended for use in medical, life saving, or life sustaining applications.
Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

The Intel® Core™ Microarchitecture, Intel® Atom, Intel® Pentium, Intel® Pentium II, Intel® Pentium III, Intel® Pentium 4, Intel® Pentium Pro, Intel® Pentium D, Intel® Pentium M , Itanium®, Xeon® may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request. Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.
This document contains information on products in the design phase of development. Do not finalize a design with this information. Revised information will be published when the product is available. Verify with your local sales office that you have the latest datasheet before finalizing a design.
All dates specified are target dates, are provided for planning purposes only and are subject to change. All products, dates, and figures specified are preliminary based on current expectations, provided for planning purposes only, and are subject to change without notice.
Intel and the Intel logo is a trademark or registered trademark of Intel Corporation or its subsidiaries in the United States and other countries.

*Other names and brands are the property of their respective owners.
Copyright © 2009, Intel Corporation
Agenda

- Introduction to Chip Design
- The Problem
- The Proposal
- Fuzzing the RTL
Agenda

- Introduction to Chip Design
- The Problem
- The Proposal
- Fuzzing the RTL
As our future becomes increasingly connected, Intel is developing advanced technologies that are enabling an entirely new line of laptops, (MIDs) Mobile Internet Devices, and more.
Sand To Silicon Video
Technology in Present Day

With >100,000,000 transistors in one IC (Integrated Circuit)
Chip Design Process

Logic Design process

Tapeout

Architectural Design

Technology Trend

Specification

Logic Design

μ-Arch

RTL

Logic Simulation

Gate Level Simulation

Logic Synthesis

Physical Design

Floor plan – P&R – Clk Tree – LVS - DRC
Register Transfer Level (RTL)

High-level representation of a circuit
Circuit behavior
- transfer of data between hardware register
- logical operation performed on the signals
2 elements – registers and combinational logic
Hardware Description Language – Verilog, VHDL

Verilog RTL Code

```verilog
if (CLK === 1'bX)
begin
    F         <= #100 {1{1'bX}};
end
else
begin
    F         <= #100 S | ( ~C & F);
end
end
```
The Problem

- Complexity: many features implemented in a single chip
- Hackers: creative, attack methods have no boundary

Bug, if escapes, could control million of gates

Does it mean an attack cannot be pre-planned and it just happens on the platform?
The Evolution

Focus RTL Testing

Fuzzing the RTL
Focus Penetration Testing

- Focus Testing
- One test per specific target

Entry Point

Access Control

Planning ➔ Discovery ➔ Attack

Reporting ➔ Additional discovery

Figure 3.1: Four-Stage Penetration Testing Methodology
Dynamic Security Testing-Fuzzing

Directed Random Testing (Fuzzing)
A group of tests targeting sub-domain

Multiple Access Control
-independent Access Control can test at same group of tests

Entry Point
Input weight

Figure 3.1: Four-Stage Penetration Testing Methodology

Planning → Discovery → Attack → Reporting
Introduction to Chip Design

The Problem

The Proposal

Fuzzing the RTL
Dynamic Security Testing (DST) Benefit - Coverage comparison

**DST**: capable to generate much higher coverage than pure focus testing

**Fuzzing Test input** to hit all of Comprehensive Attack Scenarios + Specific Attack Scenarios

**Focus Testing**: Testing on specific scenario only

**Total Security Coverage** = Comprehensive Attack Scenarios + Specific Attack Scenarios + Additional Scenarios generated from Fuzzing

**Total Security Coverage** = Specific Attack Scenarios
Agenda

- Introduction to Chip Design
- The Problem
- The Proposal
- Fuzzing the RTL
How?
Threat Model
Testing Analysis

Logic Path from Threat Agent to Asset
Domain to Test

Logic Path from Threat Agent to Asset
Partition the Testing Environment

Logic Path from Threat Agent to Asset

Access Control
Asset (Key)
Asset (data)
Threat Agent

Potential path taken by Threat Agent
What can you do to attack a design in RTL phase?
Get the Specification

Get a product specification

Relationship of the Asset, Threat Agent and Access Control according to the specification
Specification – an example

AES128_FAST

Requirement

The unencrypted/decrypted data, as well as key, are protected from the Threat Agents between the LOAD and DONE assertion.
Get the RTL code

Which RTL code used by the product?
- Available IPs?
- Proprietary IPs?

Free open source IP
http://www.opencores.org/mailman/listinfo/cores

Commercial IP

Write your own code ..., if you are interested,
- VHDL Tutorial: Learn by Example
  http://esd.cs.ucr.edu/labs/tutorial/

Note: For this presentation, the aes_crypto_core was downloaded from OpenCores (www.opencores.org), with some modification.
Get the Logic Simulator

- Open source logic simulator
  - Verilator, VeriWell, etc.

- Commercial logic simulators
  - LogicSim, ModelSim, VCS, etc.
  - Some may have free version for students

- For more complete list,

Note: For this presentation, the logic simulator used was VCS
Environment Setup for VCS

Installation path
Download from VCS ...

Source the setup file

synopsys_sim.setup file:
- list of libraries
  - common setting

```bash
> more synopsys_sim.setup
crypto_lib : ./crypto_lib
```
Analyze the design

- vhdl file
  vhdlan <file name>
- verilog file
  vlogan <file name>
- system verilog file
  vlogan –sverilog <file name>
- Other simple switches
  - -f <file contains list of design file to compile>
  - -work <target library name>

NOTE: For this presentation, VHDL code was used.
Elaboration

to run in ucli
  – vcs <top module / testbench>

to run in gui
  – vcs –debug_all <top module / testbench>
Simulation
run simulation and stop when $finish is called
  – simv
run simulation in ucli
  – simv -ucli
run simulation in dve
  – simv -gui
Summary – steps to bring up RTL Simulation

To run the simulation, at the DVE command line, dve> run 3us
Fuzzing the RTL

<table>
<thead>
<tr>
<th>Access Control</th>
<th>Threat Agent</th>
<th>Asset</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td>DFT1</td>
<td>KEY</td>
</tr>
<tr>
<td>LOAD</td>
<td>DFT2</td>
<td>DATA_IN</td>
</tr>
<tr>
<td>MODE</td>
<td></td>
<td>DATA_OUT</td>
</tr>
</tbody>
</table>

-Fuzzing at Access Control and Threat Agent (input)
-monitor the Asset

An example

<table>
<thead>
<tr>
<th>CLK</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
<th>21</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>LOAD</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>MODE</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>DFT1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>DFT2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
The Results

Protected period

BUG!!
What Next?

Fix the design
Test the design fixes until no issue found
Notes

For large design – Coverage Based Validation method is used, instead of manually examined the waveform
The Benefit ...

This method can be used in any design, if you have the specification and the RTL code.

If you are the RTL developer, this is a good method to ensure your design can withstand the attack.
Acknowledgement

Thanks to my colleagues from intel Penang Design Center, Jonie Lim, CP Teh and Thanh Le Nguyen for their contribution in this presentation.
Thank you

Yours Truly

mary.siaw.see.yeoh@intel.com