Verification Techniques for COTS Dedication of Commercial FPGA Tools

Junbeom Yoo, Eui-Sub Kim, Sejin Jung
Dependable Software Laboratory
KONKUK University

2015.06.04
Formal Verification Techniques which can be used for COTS SW Dedication of Commercial FPGA Tools used to Develop Safety-Critical Control Software in Nuclear Power Plants

Junbeom Yoo, Eui-Sub Kim, Sejin Jung
Dependable Software Laboratory
KONKUK University

2015.06.04
Platform Change from PLC to FPGA

Digital I&C (Instrumentation & Control) in nuclear power plants

**PLC (Programmable Logic Controller)** has been used to implement I&Cs for decades
- **SW development** on industrial computers (CPU & OS)
However, increasing maintenance cost and CCF (Common Cause Fault) problem in security
- Request for alternative implementation platforms

**FPGA (Field Programmable Gate Array)** is an alternative platform of PLC for I&Cs
- Higher computation performance and stronger security
- HW development

FBD program for PLC

Netlist design for FPGA
FPGA Development Process

1. Requirement or Design Specification
2. RTL Design (Verilog or VHDL)
3. Gate-Level Design (Netlist)
4. Logic Synthesis (3rd Parties)
5. Place & Route (Chip Supplier)
6. Layout
7. IDE (Chip Supplier)
8. FPGA
FPGA Development Process + Verification

1. Requirement or Design Specification
2. RTL Design (Verilog or VHDL)
3. Gate-Level Design (Netlist)
4. Layout
5. FPGA

- Property
- Model Checking
- Equivalence Checking
- Equivalence Checking
- Test Bench
- RTL Simulation
- Gate-Level Simulation
- Post-Layout Simulation
- Static Timing Analysis
COTS SW Dedication

A process for demonstrating correctness and safety of commercial software (COTS) used directly or indirectly

- Direct COTS SW : Directly **used** in an application to perform safety functions
- **Indirect** COTS SW : Directly **produces** direct SW (not COTS SW)

Two international standards to cope with for digital I&Cs in NPP

<table>
<thead>
<tr>
<th>Standards</th>
<th>Target</th>
<th>Process</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>EPRI-NP5652 (EPRI TR-106439)</td>
<td>Commercial Grade Item (CGI) + Software-based equipments</td>
<td>Method 1 ~ 4</td>
<td>Focusing on Direct CGI Base of Korean Std.</td>
</tr>
<tr>
<td>NUREG/CR-6421</td>
<td>Direct / Indirect COTS software</td>
<td>Processes for each safety category</td>
<td>Containing Indirect CGI</td>
</tr>
</tbody>
</table>
COTS software such as logic synthesis and IDEs are always used to develop FPGA.
- Indirect COTS SW & Category B
- Should take the **COTS SW dedication** process according to the standards
NP-5652 suggests 4 methods

Method 1: Special Test and Inspection
- Verifying important functionalities

Method 2: Commercial-Grade Survey
- Confirming and evaluating QA program of suppliers

Method 3: Source Verification
- Verifying critical characteristics at the supplier's facility (often impossible)

Method 4: Item/Supplier Performance Record
- Verifying acceptability through documented items or supplier's performance records

Method 1 is important for logic synthesis
- Functionality to verify: correct synthesis
- Direct compiler verification techniques can't be used
- It is a commercial compiler (No source code opened)

Indirect verification is required
- Logic Equivalence Checking (LEC) for specific inputs
Logic Equivalence Checking

Formally verify(prove) that
- for a specific input, the output always shows the same behavior with the input

Commercial LEC tools
- FormalPro (Mentor Graphics)
- Formality (Synopsys)
- Encounter Conformal EC (Cadence)
- Jasper Gold (Cadence)
- Quartz Formal (Magma Design Automation)
- 360 EC (OneSpin Solutions)
**Applicability of LECs**

Applicability depends on the tool combinations  
- LEC x Logic Synthesis x IDEs

No applicable LEC for **Synopsys Synplify Pro** (in Actel Libero IDE)  
- In this case, we need to develop a customized LEC

<table>
<thead>
<tr>
<th>Logic Synthesis</th>
<th>IDE</th>
<th>Mentor Graphics</th>
<th>Cadence Encounter</th>
<th>Synopsys Formality</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>FormalPro</td>
<td>Conformal EC</td>
<td></td>
</tr>
<tr>
<td>Mentor Graphics</td>
<td>Xilinx ISE</td>
<td>O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Precision RTL</td>
<td>Actel Libero Soc</td>
<td>O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Synopsys Synplify Pro</td>
<td></td>
<td>O</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Xilinx ISE</td>
<td>O</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Actel Libero Soc</td>
<td>O</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Altera Quartus II</td>
<td>O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Xilinx XST</td>
<td>Xilinx ISE</td>
<td>O</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>Synopsys DC Ultra</td>
<td>-</td>
<td></td>
<td></td>
<td>O</td>
</tr>
</tbody>
</table>
A New Customized LEC: CVEC (A Customized VIS based Equivalence Checking)

A VIS based solution (VIS: Verification Interacting with Synthesis)

It can verify the combination of ‘Synopsys Synplify Pro’ with ‘Actel Libero SoC’
- An open-sourced formal verification tool, VIS
- Translators requires (step1,2) to use the VIS
- Verification performance is up to the VIS

[3 Steps]
1. Verilog → Verilog4VIS
2. EDIF → BLIF-MV
3. VIS Equivalence Checking

Equivalence?

Target Synthesis Tool

The combination of ‘Actel Libero IDE’ + ‘Synopsys Synplify Pro’
Summary

**FPGA** is receiving international attention as an alternative platform of digital I&Cs in NPPs.

We should do the **COTS SW dedication** to demonstrate correctness and safety of commercial software (COTS) used indirectly, such as **FPGA logic synthesis** and **IDEs**, according to international standards.

**LEC** (Logic Equivalence Checking) is strongly suggested as a means of the special test (Method 1).

Our target (Current working set) - the combination of **Actel Libero Soc** with **Synopsys Synplify Pro** has no LEC solution applicable.

In this case, we may need to develop a new customized solution.

COTS SW dedication of indirect SW involves an in-depth analysis on the target’s functionality and the techniques used to verify the functionality.
THANK YOU
module FIX_RISING (clk, rst, pulse, RNG_E, MDL_E, AI_E, OB_INIT_STA, PTSP, TSP, P_1);

input clk;
input pulse;
input rst;

parameter [15:0] PV_OUT = 12000;
input RNG_E;
input MDL_E;
input AI_E;
input OB_INIT_STA;
output [15:0] PTSP; reg [15:0] PTSP;
output [15:0] TSP; reg [15:0] TSP;
output [15:0] TRIP_CNT; reg [15:0] TRIP_CNT;
output [15:0] PTRIP_CNT; reg [15:0] PTRIP_CNT;
output TRIP_LOGIC; reg TRIP_LOGIC;
output PTRIP_LOGIC; reg PTRIP_LOGIC;
output TRIP;
output PTRIP;
output P_1;
output clkclkclk;

parameter [15:0] HYS = 300;
parameter [15:0] PHYS = 300;
parameter [15:0] RNG_MIN = 600;
parameter [15:0] RNG_MAX = 29400;
parameter [15:0] MAXCNT = 20;

// local variable 0 is digits, skip defining a parameter
// local variable 1 is digits, skip defining a parameter
parameter TRUE = 1;
parameter FALSE = 0;