OpenFPGA: An Opensource Framework Enabling Rapid Prototyping of Customizable FPGAs

Xifan Tang, Edouard Giacomin, Aurélien Alacchi
Laboratory of Nanointegrated Systems (LNIS) – University of Utah, U.S.A

09/11/2019 – Barcelona, Spain

OpenFPGA: Context

- FPGAs’ ever-increasing role in modern computing systems
- Prototyping FPGAs is traditionally a cumbersome process
- Considerable manual layout (Groups of hardware engineers)
- Ad-hoc design tool development (Groups of software engineers)
- Year-long development cycles
- Full-Custom Flow
- OpenFPGA: The First open-source FPGA IP generator
  - Enable a rapid prototyping flow for FPGA IPs (the fabric)
  - Customizable FPGA architecture and instant bitstream support
- Automated Semi-Custom approach
- Benefit to industry (eFPGA or specific application field)
- Benefit to academia (2x smaller and 3x faster than state-of-the-art)

OpenFPGA: How is it different?

- VTR (Verilog To Routing):
  - Architecture exploration tool
- SymbiFlow: SymbiFlow
  - Open source flow to generate bitstream from Verilog
  - Target commercial FPGA
- OpenFPGA: OpenFPGA
  - Customizable architecture
  - Bitstream related to the specified architecture
  - Each tool has its specific application

OpenFPGA: Flow

Automate FPGA development using a semi-custom design approach

XML Hardware Description Extension

Compared hierarchy
Low level customization
Basic element composition
Multimode support

OpenFPGA

<architecture>
  <models>
    <device>
      <switchlist>
        <segmentlist>
          <complexblocklist>
            <pb_type>
              <mode>
                <interconnect>
                  <power>
                    <clocks>

Verilog Benchmark
Netlist
XML-based FPGA Architectures Description
Standard/Customized Cell Library
Verification Flow
Formal Verification
Formal Tool
Verification
<architecture>
  <models>
    <device>
      <switchlist>
        <segmentlist>
          <complexblocklist>
            <pb_type>
              <mode>
                <interconnect>
                  <power>
                    <clocks>
Circuit Level Customization

XML definition for a MUX to be auto-generated in Verilog:

```xml
<design_technology type="cmos" structure="one-level"/>
<port name="Q" size="1"/>
<port name="CLK" size="1"/>
<port name="D" size="1"/>
...</design_model>
```

XML definition for a FF to be mapped to standard or customized cell:

```xml
<design_technology type="cmos" structure="one-level"/>
<port name="Q" size="1"/>
<port name="CLK" size="1"/>
<port name="D" size="1"/>
...</design_model>
```

Composition of Basic Elements

- Multiplexers can have one-level, multi-level or tree-like structure

![Multiplexer Diagram](image)

Multimode support overview

Example of 6-modes BLE with adders and fracturable LUTs

- Separated XML description for different modes of a CLB
- Verilog and Bitstream will be automatically generated based on the physical implementation

![Multimode Diagram](image)

Multimode definition in XML

- Physical mode
  ```xml
  <pb_type name="frac_logic" num_pb="1">
  <!-- ports definition -->
  </pb_type>
  ```

- Operating mode
  ```xml
  <pb_type name="frac_lut6" num_pb="1">
  <!-- ports definition -->
  </pb_type>
  ```

Experimental Methodology

- Technology: Commercial 40 nm
- Homogeneous FPGA Architecture:
  - F = 3
  - \( F_{\text{V}} = 0.055 F_{\text{out}} = 0.1 \)
  - 1 Tile = 1 CLB + 1 SB + 2 CB
  - 1 CLB = 10 FLE
  - 1 FLE = 2 \times (LUT6 + LUT4) + 2 \times 1-bit adder + 2 \times FF
  - Channel width = 300: 83% length-4 + 13% length-16
- Chip utilization rate = 80%
- Baseline:
  - Current state of the art [1]
  - StratixIV + QuartusII

Cell Optimization

- Cell library has a strong impact on the FPGA metrics: 90% of the FPGA area is made of CCFF and multiplexers
- Tgate-based MUX2 is 2.5x smaller than from the SC library
- CCFF is 1.8x smaller than from the SC library

Few custom cells in a semi-custom flow can make the difference

Area Analysis

- Post-layout evaluation using a commercial 40nm node and a Stratix IV-like architecture
- Flow runs in 24h
- Using an optimized cell library (multiplexers, CCFF):
  - 1.8x area reduction

<table>
<thead>
<tr>
<th>Path Type</th>
<th>Previous</th>
<th>OpenFPGA (Std Cells)</th>
<th>OpenFPGA (1/2-lvl MUXes)</th>
<th>Stratix IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>5-LUT</td>
<td>0.46</td>
<td>0.14</td>
<td>0.27</td>
<td>0.26</td>
</tr>
<tr>
<td>6-LUT</td>
<td>0.5</td>
<td>0.15</td>
<td>0.27</td>
<td>0.28</td>
</tr>
<tr>
<td>1-bit Adder</td>
<td>0.7</td>
<td>0.54</td>
<td>1.00</td>
<td>0.77</td>
</tr>
<tr>
<td>20-bit Adder</td>
<td>1.63</td>
<td>1.10</td>
<td>2.12</td>
<td>1.23</td>
</tr>
<tr>
<td>Local Routing</td>
<td>0.27</td>
<td>0.12</td>
<td>0.59</td>
<td>0.59</td>
</tr>
<tr>
<td>L-4 track</td>
<td>2.53</td>
<td>0.40</td>
<td>0.75</td>
<td>0.59</td>
</tr>
<tr>
<td>L-16 track</td>
<td>4.02</td>
<td>0.76</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Timing Analysis

- Post-layout evaluation using a commercial 40nm node and a Stratix IV-like architecture
- Flow runs in 24h
- Using an optimized cell library (multiplexers, CCFF):
  - 3x delay reduction

<table>
<thead>
<tr>
<th>Path Type</th>
<th>Previous</th>
<th>OpenFPGA (Std Cells)</th>
<th>OpenFPGA (1/2-lvl MUXes)</th>
<th>Stratix IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>5-LUT</td>
<td>0.46</td>
<td>0.14</td>
<td>0.27</td>
<td>0.26</td>
</tr>
<tr>
<td>6-LUT</td>
<td>0.5</td>
<td>0.15</td>
<td>0.27</td>
<td>0.28</td>
</tr>
<tr>
<td>1-bit Adder</td>
<td>0.7</td>
<td>0.54</td>
<td>1.00</td>
<td>0.77</td>
</tr>
<tr>
<td>20-bit Adder</td>
<td>1.63</td>
<td>1.10</td>
<td>2.12</td>
<td>1.23</td>
</tr>
<tr>
<td>Local Routing</td>
<td>0.27</td>
<td>0.12</td>
<td>0.59</td>
<td>0.59</td>
</tr>
<tr>
<td>L-4 track</td>
<td>2.53</td>
<td>0.40</td>
<td>0.75</td>
<td>0.59</td>
</tr>
<tr>
<td>L-16 track</td>
<td>4.02</td>
<td>0.76</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Where to progress?

- **MCNC big20 suite benchmarks**
  - Average of 35% critical path reduction compared to previous work
  - Average of 45% (30 + 15) critical path overhead compared to Stratix-IV
- Gap between academic and commercial EDA tools

Critical paths comparison

Previous work [9]

- OpenFPGA + 1/2-lvl MUXes + Opt.Cells
- Stratix-IV

Critical paths comparison


Conclusion

Acknowledgment

This material is based on research sponsored by Air Force Research Laboratory (AFRL) and Defense Advanced Research Projects Agency (DARPA) under agreement number FA8650-18-2-7855. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Air Force Research Laboratory (AFRL) and Defense Advanced Research Projects Agency (DARPA) or the U.S. Government.

Pierre-Emmanuel Gallardon and Xifan Tang have financial interests in the company ReRouting LLC, which manufactures RRAM-based systems and provides engineering service.

OpenFPGA: Summary

- Fully functional XML to Prototype flow (FPGA-X2P) supporting homogenous multi-mode FPGA fabrics
  - XML-to-Verilog generator
  - Verilog-to-Bitstream generator
  - Verilog testbenches for functionality/formal validation
- Automatic backend flow for homogenous FPGA
  - 20×20 FPGA layout using a commercial 40nm node in 24h
  - 2× area and 3× performance improvement over previous arts
- OpenFPGA alpha release with tutorials
  - GitHub repository: https://github.com/LNIS-Projects/OpenFPGA
  - Online documentation: https://openfpga.readthedocs.io/en/master/

Thank you for your attention

Questions?