Incremental Signal-Tracing for FPGA Debug

Eddie Hung, Steven J. E. Wilton
{eddieh, stevew}@ece.ubc.ca

University of British Columbia
Vancouver, Canada

FPL :: Aug 2012
What this talk is about

- FPGAs have many advantages
  - Ability for “Desktop Fabrication”
  - Commonly used to prototype and verify ASICs
What this talk is about

- FPGAs have many advantages
  - Ability for “Desktop Fabrication”
  - Commonly used to prototype and verify ASICs
- But even so, debug is still hard!

Diagram:
- Design → Test → Error → !?!
What this talk is about

• FPGAs have many advantages
  • Ability for “Desktop Fabrication”
  • Commonly used to prototype and verify ASICs

• But even so, debug is still hard!

• This work:
  Incremental techniques for trace-buffer insertion
Introduction

- Debug is the process of locating and eliminating design errors – 'bugs' – in ICs
- Important as mistakes in silicon cost big money
  - 2007: AMD K10 TLB bug – 4 months
  - 2011: Intel 'Sandy Bridge' chipset – $700 million
Introduction

• Pre-Silicon techniques alone are insufficient
  • Software simulation effective, but slow
    - Latest Core i7 (2.6 GHz) simulates at 2-3Hz
  • Formal verification limited to small components
  • Unable to interact with real-world stimulus

• FPGA prototypes -- fast and physical
  • Instant circuit fabrication: quick turnaround
  • Runs at near-speed: increased coverage
Introduction

- For debug, same challenge as ASICs: visibility
  - Limited I/O: lack of access to internal nodes
Introduction

- Enhance observability with trace-buffers:
  - Sample a subset of signals into on-chip memories
  - Capturing a sequence of states, at full speed
Introduction

- Enhance observability with trace-buffers:
  - Sample a subset of signals into on-chip memories
  - Capturing a sequence of states, at full speed
  - Does not cost extra silicon area
    - FPGAs commonly not filled to capacity
- Example IP: Xilinx ChipScope, Altera SignalTap, Tektronix Certus
Incremental-Tracing

⇒ Faster debug turnaround
Incremental-Tracing

- FPGAs are getting larger, so is CAD runtime
  - Hours or even a full day are not uncommon

![Diagram](image-url)
Incremental-Tracing

- FPGAs are getting larger, so is CAD runtime
  - Hours or even a full day are not uncommon
- Use Incremental-Compilation techniques!
Incremental-Tracing

- Incremental Compilation is not new...
- Crucially: during debug, only want to observe
- Instrument *without* modifying user-circuit
  - Incrementally add trace connections using spare resources not used in the original circuit mapping
  - Faster turnaround time between debug iterations
  - Preserve circuit mapping and avoid heuristic CAD
- But what are its limitations?
Incremental-Tracing CAD

- Two techniques to improve feasibility:
  - Many-to-many flexibility – connect to any trace-pin
Incremental-Tracing CAD

- Two techniques to improve feasibility:
  - Many-to-many flexibility – connect to any trace-pin
Incremental-Tracing CAD

- Two techniques to improve feasibility:
  - Many-to-many flexibility – connect to any trace-pin
Incremental-Tracing CAD

• Two techniques to improve feasibility:
  • Many-to-many flexibility – connect to any trace-pin
Incremental-Tracing CAD

- Two techniques to improve feasibility:
  - Many-to-many flexibility – connect to any trace-pin
Incremental-Tracing CAD

- Two techniques to improve feasibility:
  - Many-to-many flexibility – connect to any trace-pin
Incremental-Tracing CAD

- Two techniques to improve feasibility:
  - Many-to-many flexibility – connect to any trace-pin
  - Logic element symmetry – leave from any OPIN
Incremental-Tracing CAD

- Two techniques to improve feasibility:
  - Many-to-many flexibility – connect to any trace-pin
  - Logic element symmetry – leave from any OPIN

Local Logic Elements A and B can be swapped for free!
Limits to Incremental-Tracing

- Investigate using VPR6 (VTR)
  - Heterogeneous architecture with hard-mul & RAM
  - Perform packing-placement-routing as normal
  - Randomly instrument results with trace-buffers
Limits to Incremental-Tracing

- OPINs *individually* routable to any trace-pin
Limits to Incremental-Tracing

- Percentage of selections *completely* trace-able

or 1200 (2963 6LUTs)

Solid Lines: **With** LE symmetry

Dotted Lines: **Without** LE symmetry

Signals traced, normalized to memory capacity
Limits to Incremental-Tracing

- Percentage of selections *completely* trace-able

LU8PEEng (21954 6LUTs)

*Dotted lines are without LE symmetry*
Limits to Incremental-Tracing

● Runtime (75% trace demand):

![Graph showing incremental-trace runtime for different benchmarks and W_min inflation levels.](image)
Limits to Incremental-Tracing

- Effect on Critical Path Delay

<table>
<thead>
<tr>
<th></th>
<th>Probability Affected</th>
<th>Average Increase</th>
</tr>
</thead>
<tbody>
<tr>
<td>mkSMAdapter4B</td>
<td>11.9%</td>
<td>3.4%</td>
</tr>
<tr>
<td>or1200</td>
<td>4.7%</td>
<td>3.6%</td>
</tr>
<tr>
<td>mkDelayWorker32B</td>
<td>2.4%</td>
<td>3.6%</td>
</tr>
<tr>
<td>LU8PeeEng</td>
<td>&lt;0.1%</td>
<td>0.2%</td>
</tr>
</tbody>
</table>
Conclusion

- FPGAs are increasingly being used for debug
- Incremental-Tracing \(\Rightarrow\) *faster debug turnaround*
  - Circuit signals can be connected to any trace-pin: many-to-many flexibility
  - 99.4% of OPINs can be incrementally-connected to 75% of the free on-chip memory ...
  - ... an order of magnitude quicker than re-compiling, for only a 20% increase in channel width
- Code available at http://ece.ubc.ca/~eddieh