XILINX
ALL PROGRAMMABLE™

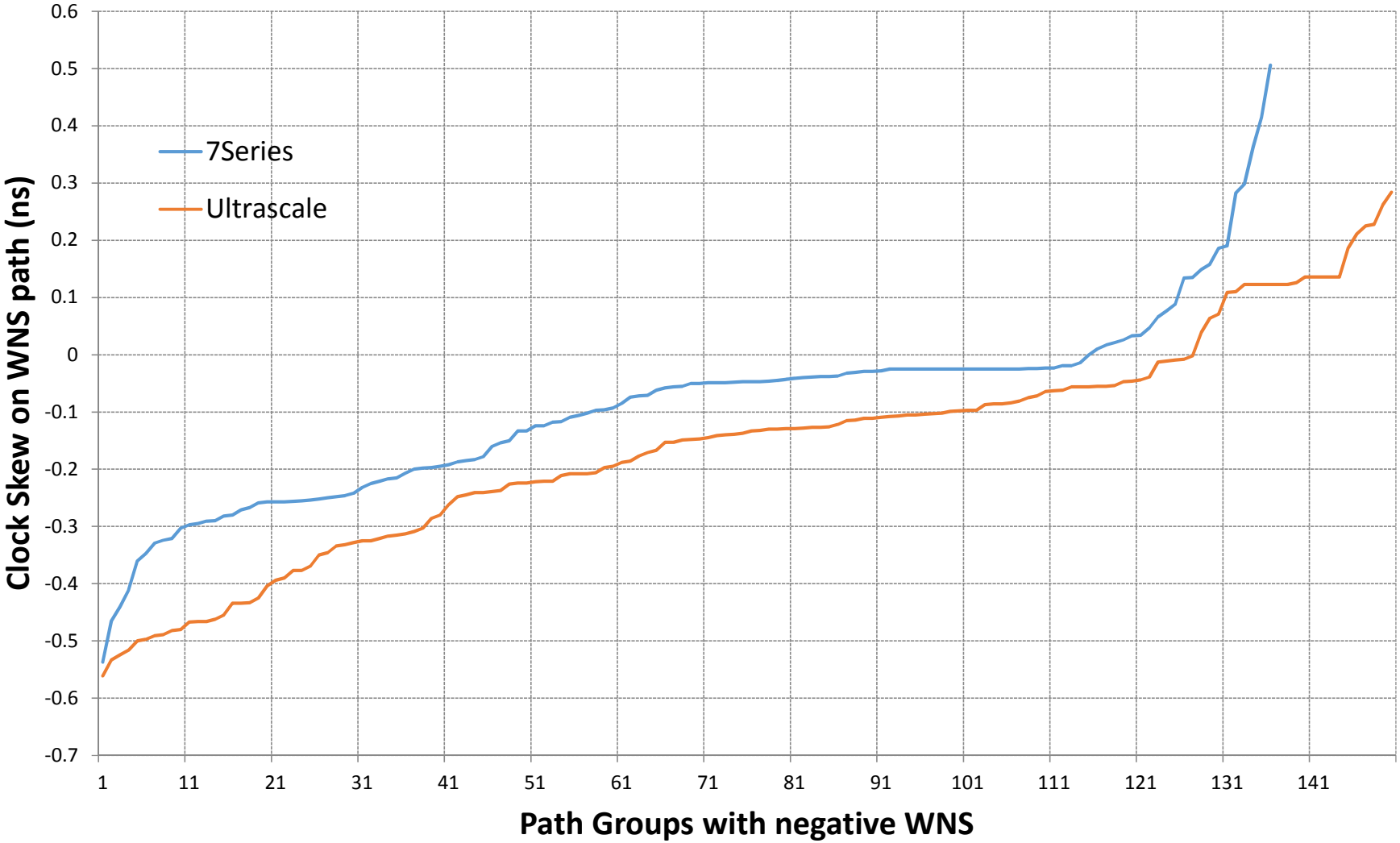
A Practical Approach to FPGA Clock Skew Optimization

Atul Srinivasan

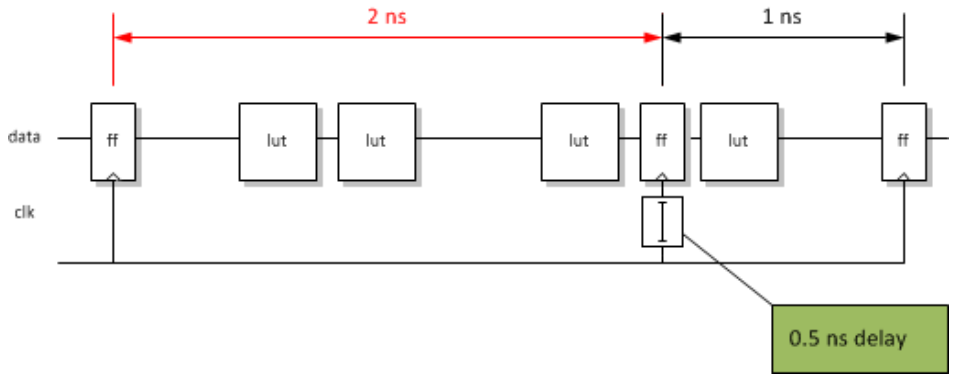
Agenda

- Acknowledgements
- Clock Skew Trend
- Project Requirements
- New Hardware Features
- Software Solution and Results
- Summary

Clock Skew Trend



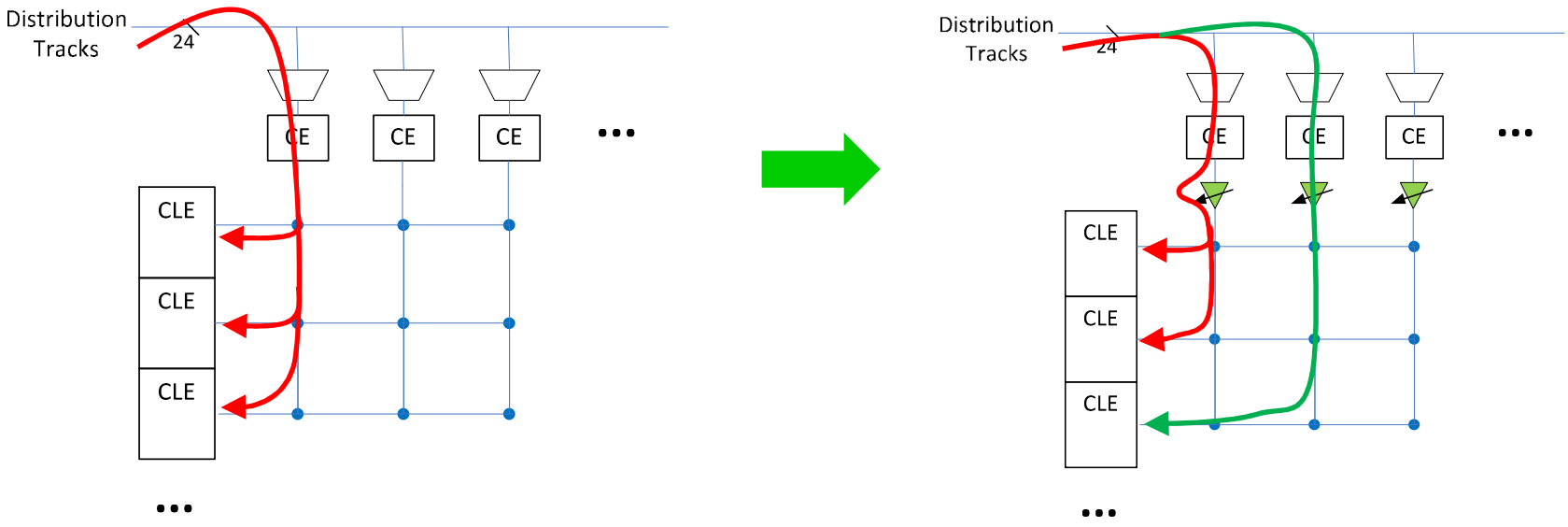
Project Requirements



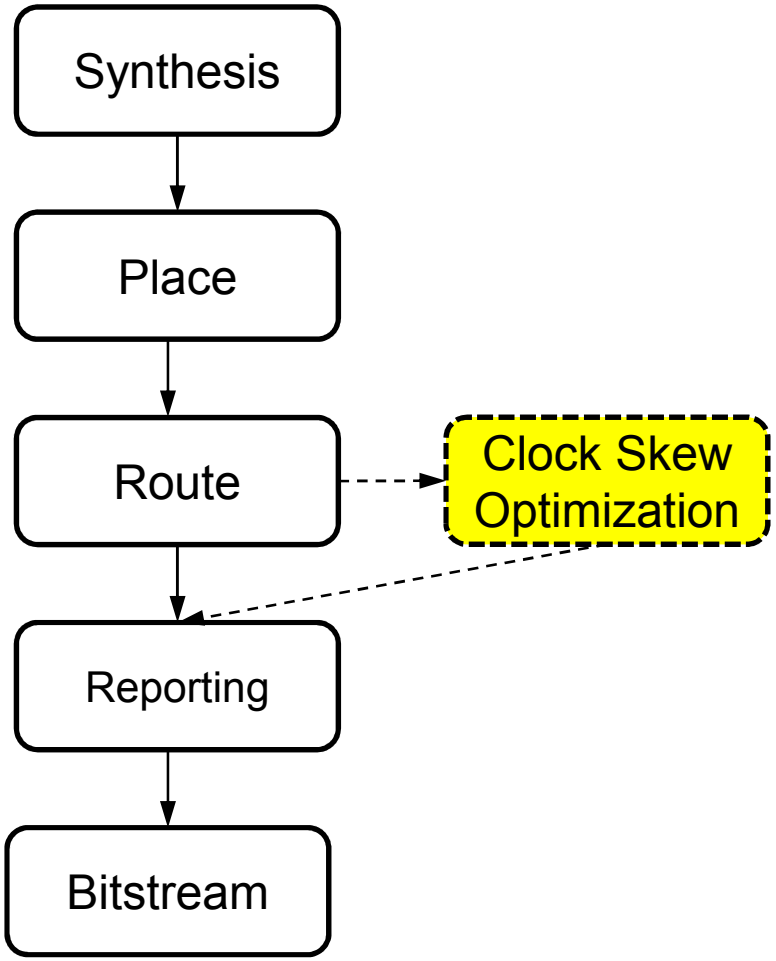
- Improve Fmax !!!
  - Validate improvement using post route Signoff Timing
- Minimally disrupt P&R flow
  - Target customer ease of use
  - Optimize for run time and memory usage

New Hardware Features

- Configurable Logic Element (CLE) leaf clock has
  - Discrete programmable delays
  - Pulse generator

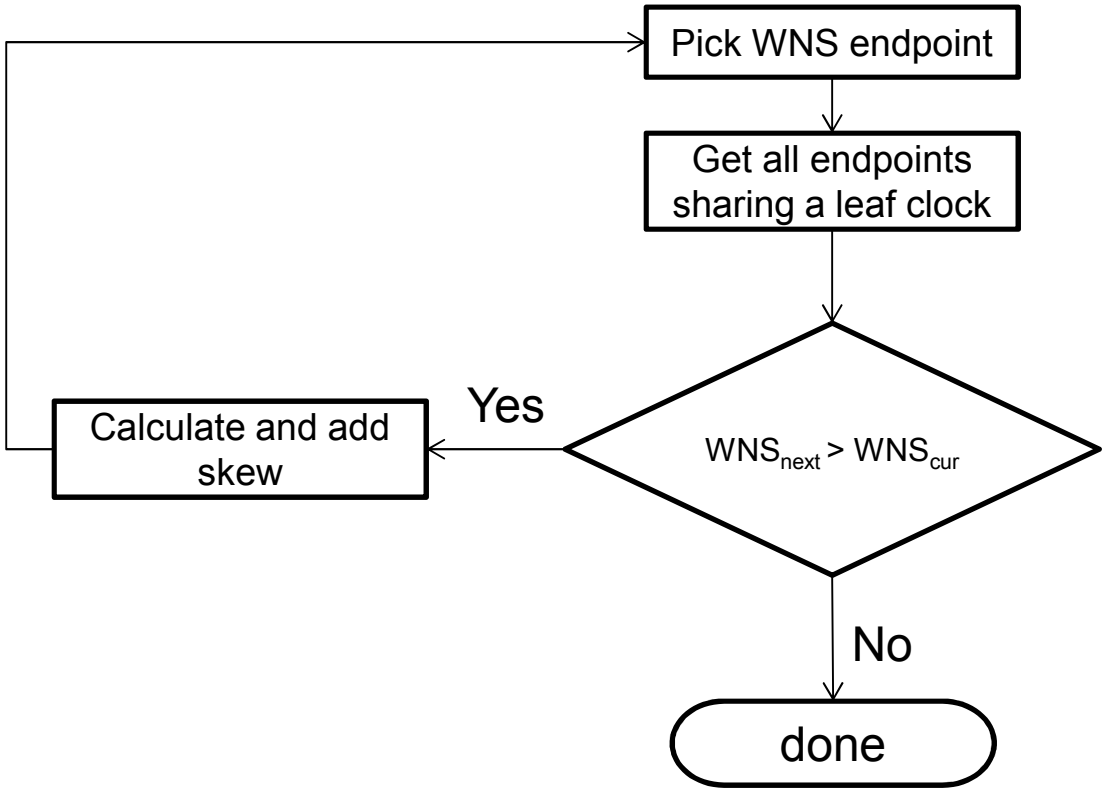


Software Flow



- **Post-route optimization**
  - Targeted for WNS
  - Could also be used to fix hold
- **Different strategies possible**
  - Fast runtime, lower Fmax: do not violate hold
  - Slower runtime, higher Fmax: violate hold and fix with hold router
- **Considered algorithms**
  - Local greedy optimization
  - Globally optimal LP solution

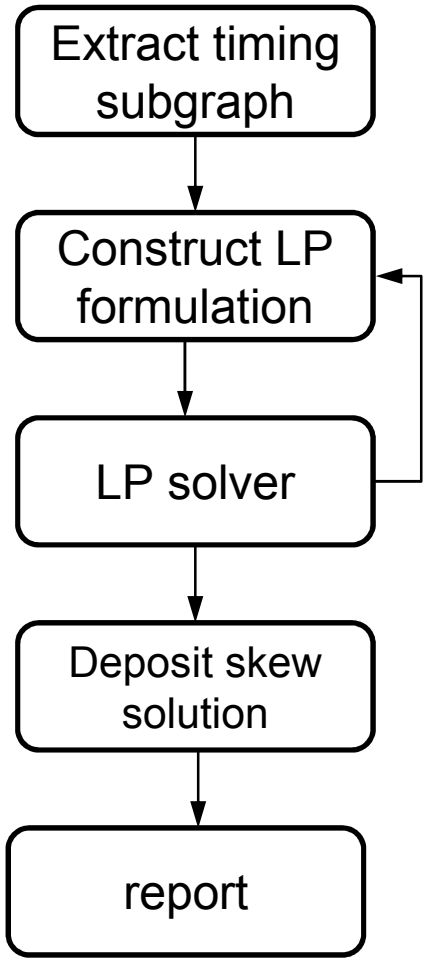
Local Greedy Algorithm Test



Runtime	Fmax increase
1-10 minutes	1-2%

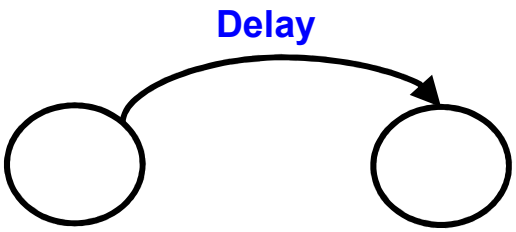
Global LP Algorithm Test

Runtime	Fmax increase
Hours	4%



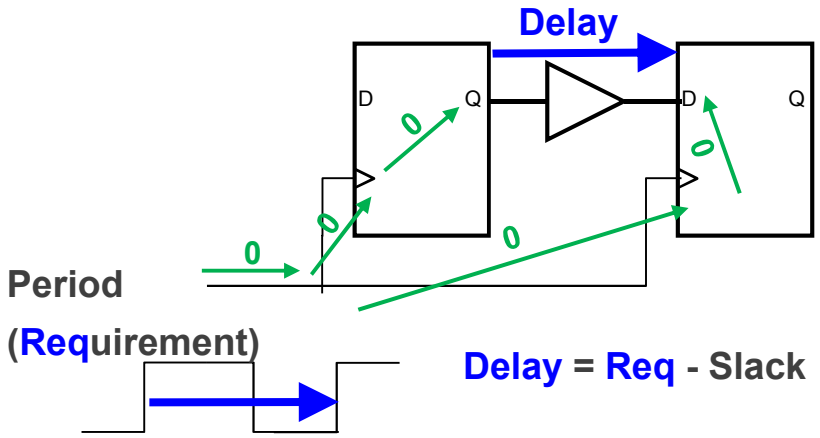
- **Extracting timing sub-graph**
  - Max paths $WNS < WNS_{worst} + 2 \times Pskew_{max}$
  - Min paths $WHS < Pskew_{max}$
  - Extracted by path enumeration
    (report_timing –nworst …)
- **Construct LP constraints for each path**
  - $PathDelay = Requirement - Slack$
  - Setup: $PathDelay - (skew_{end} - skew_{start}) < T$
  - Hold: $PathDelay - (skew_{end} - skew_{start}) > 0$
  - Objective function: $Minimum(T)$
- **LP solver**
  - Iterative process due to quantized skew values

Flop Graph



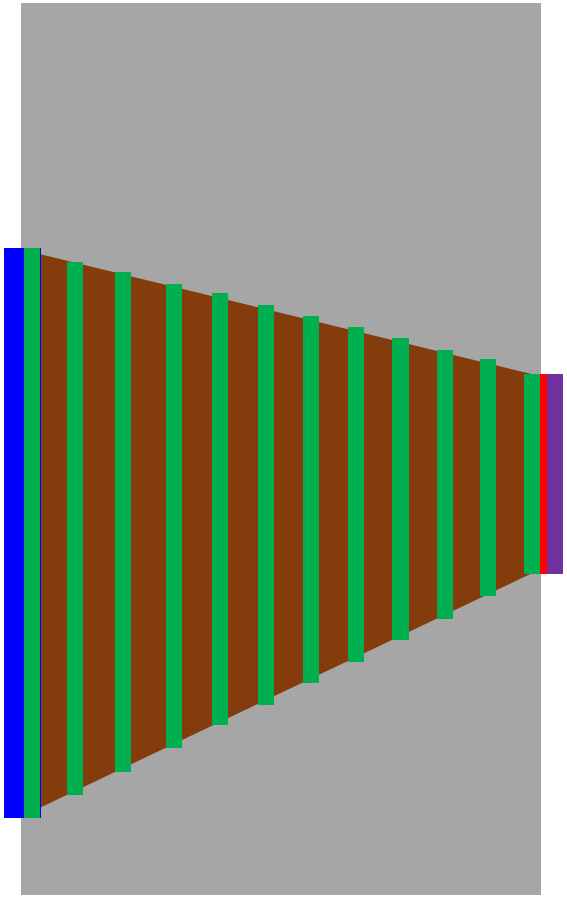
- **Graph of timing critical region**
  - Nodes = Registers and I/Os (No LUTS)
  - Edges = Worst timing paths

- **Abstraction of timing complexity**
  - Signoff accuracy
  - Handles
    - Exceptions
    - Multiple clocks
    - Latencies, Uncertainties
    - CRPR
    - Preset/Recovery timing arcs
    - Multi die analysis
    - Overlapping clock / data networks

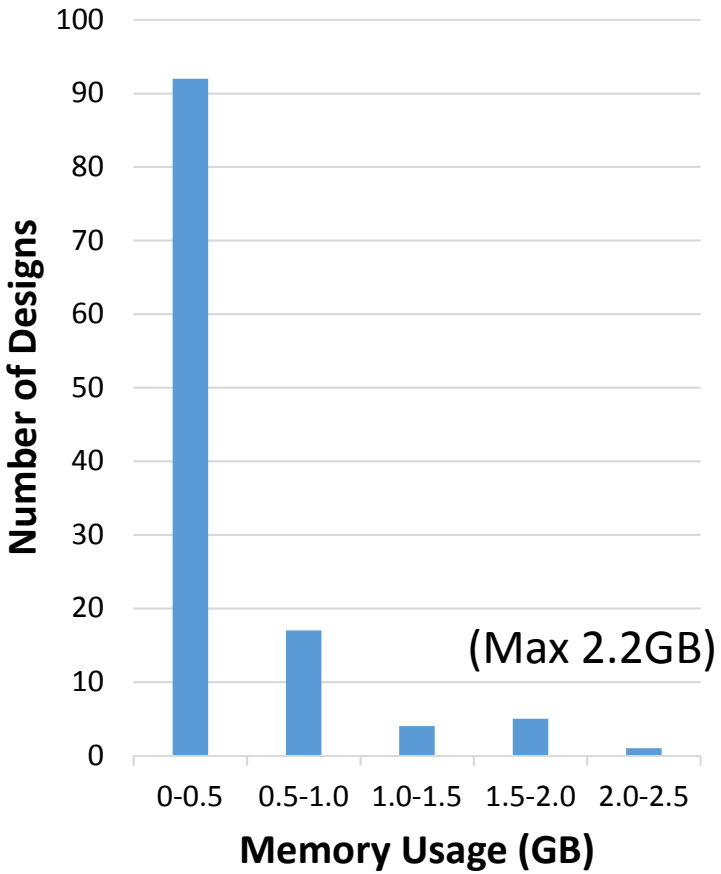


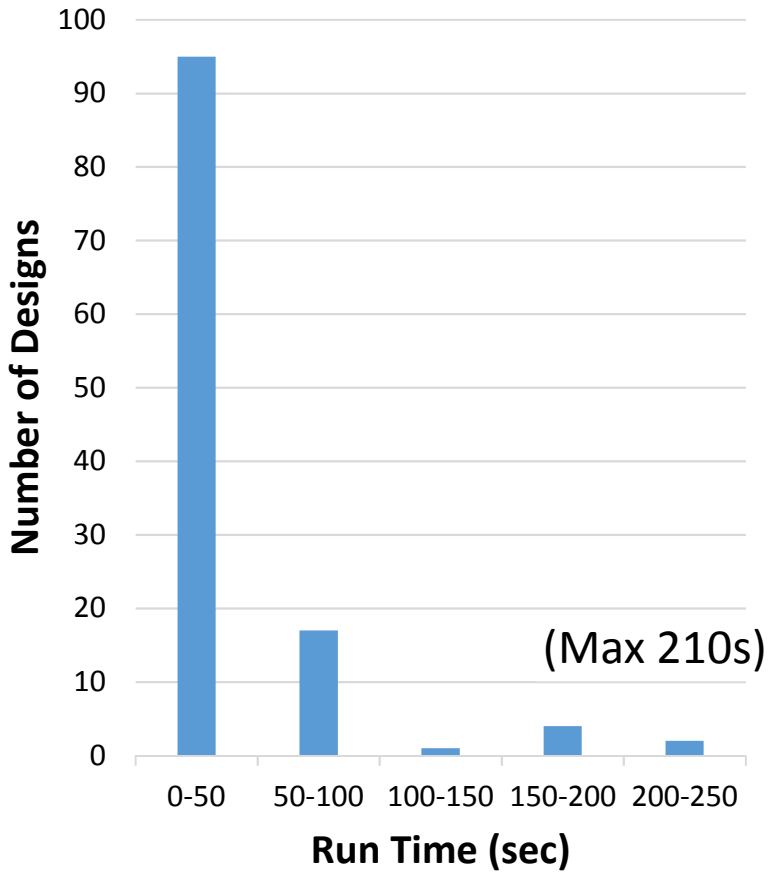
Flop Graph Computation

- **Steps**
  - Identify critical endpoints
  - Trace and mark fanin cone
  - Collect critical startpoints
    - Exclude startpoints outside critical slack range
  - Launch unique tags from each critical startpoint
  - Propagate arrivals to critical endpoints
    - Do not merge tags from different critical startpoints
    - Restrict propagation to marked cone
    - Free arrivals as soon as they are no longer needed
  - Evaluate timing checks at critical endpoints
  - Flop graph is now available from data local to critical endpoints



Memory Usage / Run Time





Pulsed Latch Extensions

Runtime	Fmax increase
<1hr	4 + 1.5%

Majority of time in LP solver

- **Create variables**
  - $xbool$ – does latch borrow
  - $xdly$ – borrow value for latch
- **Constrain maximum pulse width**
  - $xdly \leq \text{MAX\_PULSE\_WIDTH}$
- **Ensure boolean variable is set if latch borrows**
  - $xdly - xbool \geq 0$
  - $xdly - MaxCoeff * xbool \leq 0$
- **Setup path between latch candidates i → j**
  - $skew_j + xdly_j + xbool_j * \Delta setup_j - skew_i + xbool_i * \Delta c2q_i - xdly_i + T \geq PathDelay$
- **Hold path between latch candidates i → j**
  - $-skew_i + xbool_i * \Delta c2q_i + skew_j + xbool_j * \text{MAX\_PULSE\_WIDTH} \leq PathDelay$

Summary

- Presented an FPGA clock skew optimization approach which:
  - Leverages programmable delay elements and pulse generators in hardware
  - Uses an efficient timing model that abstracts STA complexity into a register graph
  - Implements an LP based global optimization engine to add beneficial skew and convert selected flip flops to pulsed latches.
- On a suite of medium to large customer designs this approach delivers a geomean Fmax improvement of 5.5% with reasonable run time and memory usage.