Optimizing Non-Monotonic Interconnect using Functional Simulation and Logic Restructuring

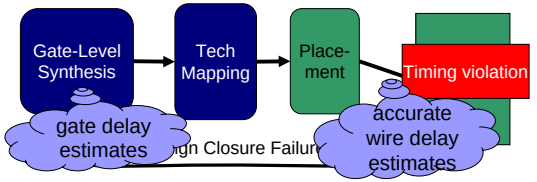
Stephen M. Plaza
Igor L. Markov
Valeria Bertacco

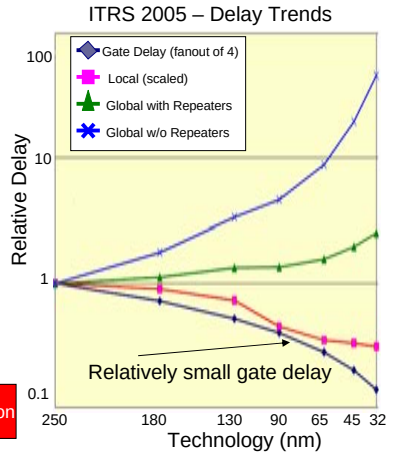
Univ. of Michigan, EECS

Increasing Significance of Interconnect Delay

70% of critical path delay is in wires

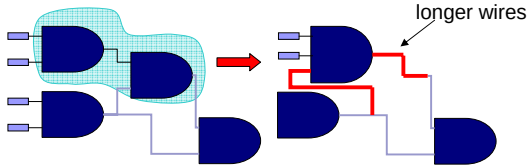
- Wire delays dominate critical paths (130nm,90,65,...)
- Gate delay no longer dominates
  - Traditional logic optimization (for gate delay only) is inadequate
- Optimize for wire delay
- Obstacles/challenges
  - Hard to predict ***accurately*** early in design flow
  - Timing closure failure—requires re-optimization





Recent Trends: Improve Early Timing Estimates

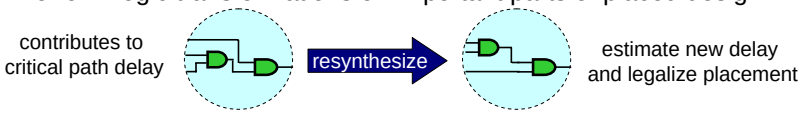
- Early global transformations can affect timing greatly
- However, interconnect delay is increasingly hard to predict early
  - e.g., technology mapping can affect wirelength



- Physical synthesis—optimization after placement
- Replication and fanout assignment, buffering
  - Area and power costs

Resynthesis after Placement

- **Minimize changes to current placement while minimizing delay**
  - More predictable delay estimates
  - Several changes could result in a costly re-iteration of placement
- Perform logic transformations on important parts of placed design



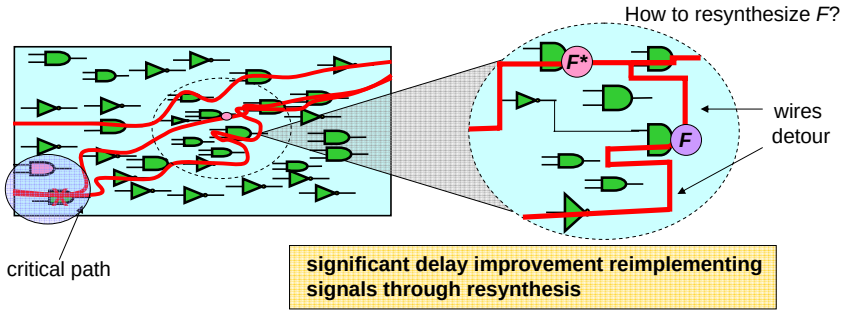
- Correct-by-construction logic synthesis strategies are limited
  - Several resynthesis implementations and topologies possible
  - Exploiting global circuit don't-cares is computationally expensive
- **Our solution**
  - Identify long wires amenable to greatest improvements
  - Use simulation-based approximations to efficiently find several quality resynthesis *candidates*
  - Find circuit don't cares w/simulation to find better candidates

Improving Wire Delay through Resynthesis

Step 1: identify long paths

Step 2: resynthesize/ restructure paths

Step 3: legalize and re-identify critical paths



significant delay improvement reimplementing signals through resynthesis

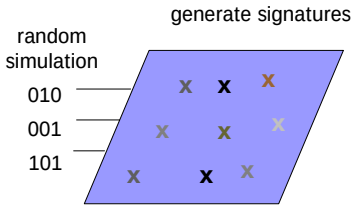
Outline

- Background: functional approximation and bit-parallel simulation
- Identifying long/non-monotonic interconnect
- Resynthesis with signature abstractions
- Experiments and Conclusions

Efficient Resynthesis through Simulation-based Approximations

- Simulation can be used to approximate circuit behavior
  - Estimate average case behavior
  - Linear-time computation

- Apply values, e.g., random, to primary inputs
- Associate *signatures* with each node
- Use signatures to enable complex optimizations

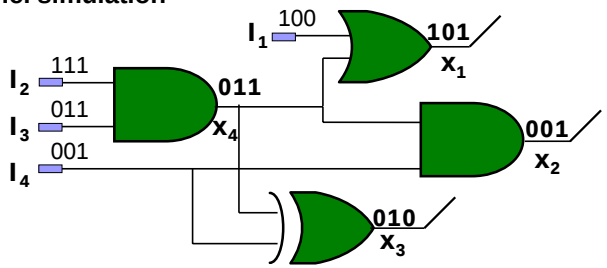


Signatures and Bit Simulation

- **Signature**: partial truth table associated with each node in a circuit
- Stimulate inputs with simulation vectors

bit-parallel simulation

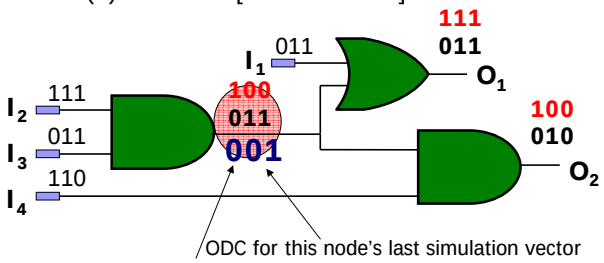
Sig(x_1) = {101}



Deriving Global Observability Don't Cares (ODCs) using Simulation

- Satisfiable don't cares (inputs that don't occur) implicitly computed
- Compute *ODC signature* for each node
- Naïve algorithm: O(n) for one node
  O(n^2) for circuit
- Fast approximation: O(n) for circuit [ASP-DAC '06]

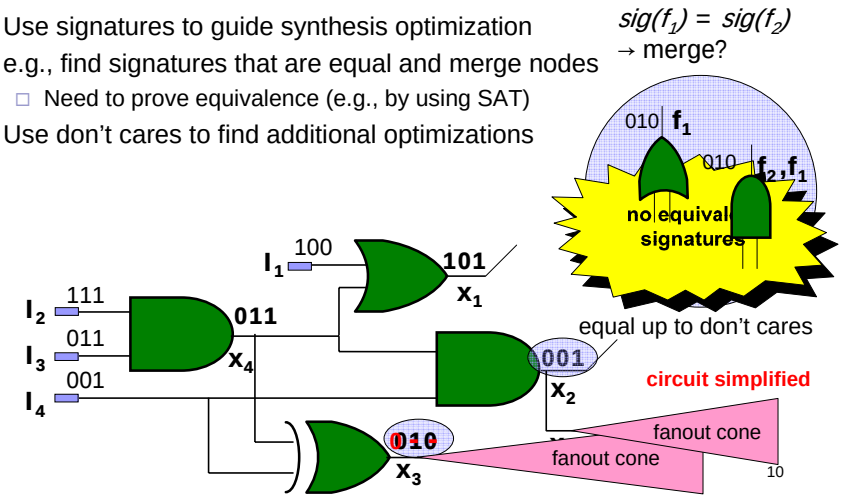
1. Invert node's signature
2. Propagate differences
3. ODC for a given simulation vector where no difference occurs at any output



Find ODCs for this node for the 3 input vectors

Simulation-based Optimization

- Use signatures to guide synthesis optimization
- e.g., find signatures that are equal and merge nodes
  - Need to prove equivalence (e.g., by using SAT)
- Use don't cares to find additional optimizations

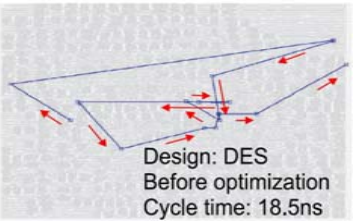


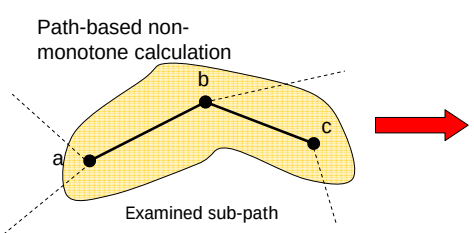
Outline

- Background: functional approximation and bit-parallel simulation
- Identifying long/non-monotonic interconnect
- Resynthesis with signature abstractions
- Experiments and Conclusions

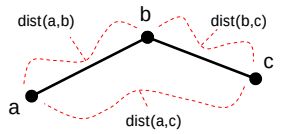
Identifying Non-Monotone Paths

- Critical paths can contain loops, detours
- Find non-monotone path segments
- Non-Monotone-Factor (NMF)
  - NMF > 1 → non-monotonicity
  - Consider sub-paths ≥2 hops
  - Dynamic programming based calculation





$$\text{NMF} = \frac{\text{dist(a,b)} + \text{dist(b,c)}}{\text{dist(a,c)}}$$



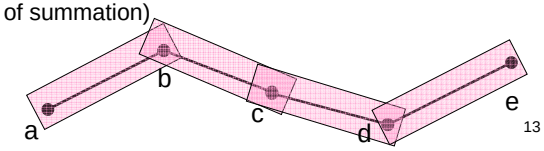
(dist = rectilinear distance)

Calculating NMF for K-Hop Paths

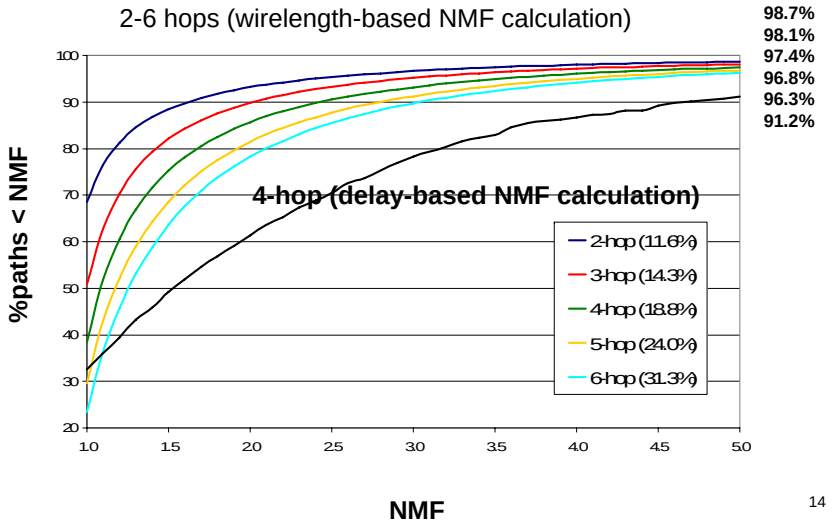
$$NMF(x_1, x_k) = \frac{1}{c_{ideal}(x_1, x_k)} \sum_{n=1}^{k-1} c(x_n, x_{n+1})$$

- C_{ideal} : ideal *cost* between two nodes
- Two metrics considered for cost
  - Rectilinear distance between nodes: optimize path length
  - Optimally buffered wire delay between nodes: optimize delay
- Dynamic programming algorithm
  - Iterate through each node of circuit
  - Explore paths K-hops from node in topological (storing sub-results of summation)

NMF(a,c) NMF(b,d)
NMF(a,d) ⋮
NMF(a,e)



Non-Monotone Paths

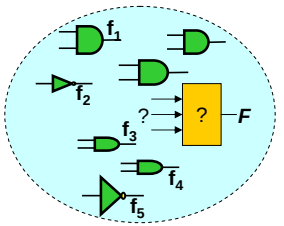


Outline

- Background: functional approximation and bit-parallel simulation
- Identifying long/non-monotonic interconnect
- Resynthesis with signature abstractions
- Restructuring of critical paths
- Experiments and Conclusions

Complexity of Resynthesis

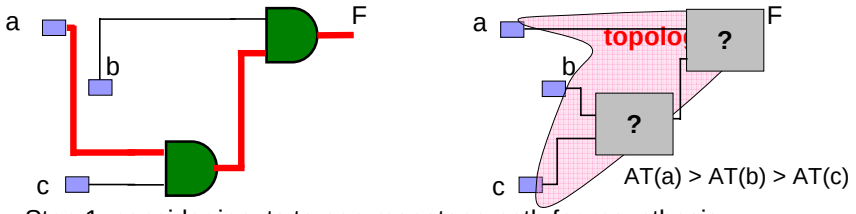
- Resynthesis: re-implement ***target*** signal *F* as a function of ***base*** signals $f_1, f_2, \ldots$
  - What base signals to consider?
  - How to construct a sub-circuit *F** with these base signals as inputs that implements *F*?



- Complexity
  - Proving if *F* can be implemented by a certain set of base signals
  - Proving that *F* = *F** (e.g., proof by construction or equivalence checking)

Finding Optimal Restructurings

find topology that will give best delay



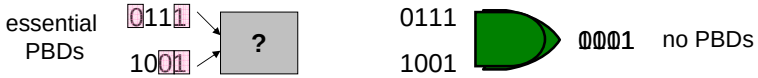
- Step 1: consider inputs to non-monotone path for resynthesis
- Step 2: find *optimal* topology candidate
  - Gives earlie
  - Greedy alg
- Step 3: find if
  - Find a set c
  - Try all combinations 2-input gate over topology

Computationally Expensive
- large topologies
- complex restructurings

Determing Logical Feasibility with Signatures

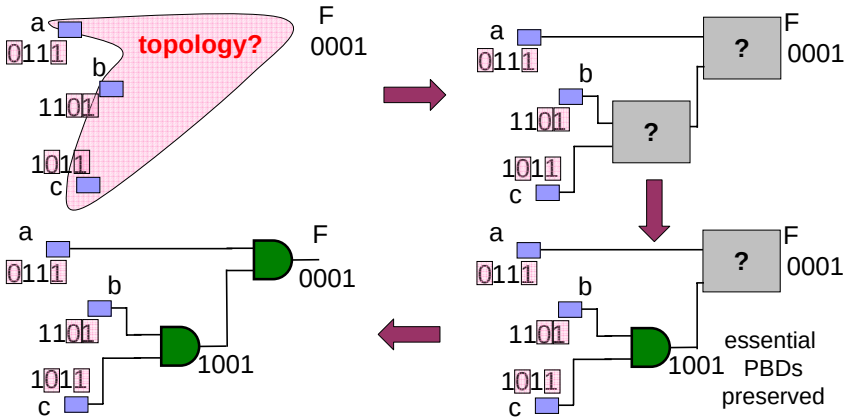
- Set of input signatures can generate an output signature if every pair of bits in the output is distinguished by at least one input [Chang '06]
  - PBDs (pairs of bits to be distinguished)

- *Essential PBDs*: PBDs distinguished by only one input signature
- For a fanout-free topology with *n* nodes and *K* simulation vectors
  - Every gate must preserve all of its inputs essential PBDs



  - Logic feasibility can be determined $O(n^2K)$ time [see paper for proof]
  - Much faster in practice, some topologies invalidated immediately

PBD-Guided Resynthesis Example



- Verify functional correctness and legalize placement
- Incrementally update timing and regenerate NMFs

Outline

- Background: functional approximation and bit-parallel simulation
- Identifying long/non-monotonic interconnect
- Resynthesis with signature abstractions
- Experiments and Conclusions

Experimental Setup

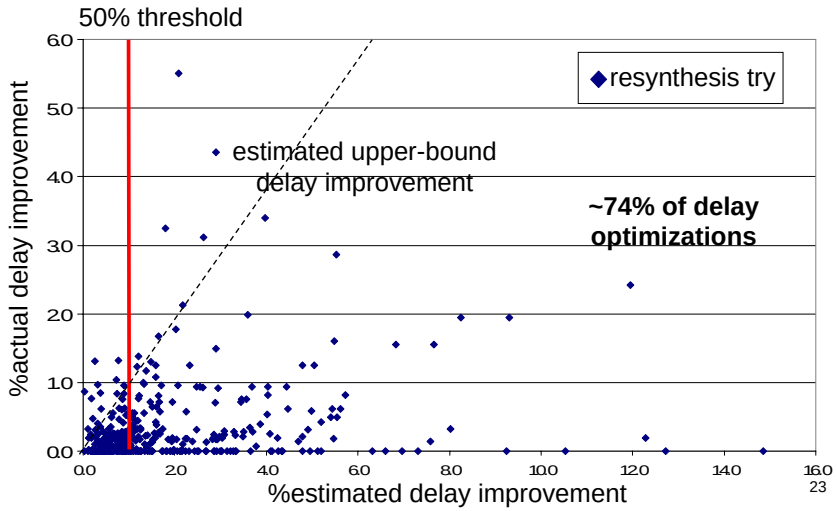
- IWLS OpenCores benchmarks
- Bit-parallel simulation (64 simulation vectors in parallel)
- SAT-based verification with MiniSAT
- Circuits placed using Capo 10
  - Several different initial placements generated for each benchmark
- STA performed using D2M delay metric
  based on Steiner trees developed by FLUTE
- Wire and gate characterizations based on 180nm technology

Delay Optimization

circuits	cell count	% delay improvement	%cell increase	time(s)
tv80	7161	9.1	0.17	1075
s38417	8278	11.7	-0.21	481
mem_ctrl	11440	9.2	-0.02	678
DMA	19118	14.5	0.08	845
aes	20795	6.4	0.01	603

- Results consistent over different initial placements
- Low overhead for corresponding delay improvement

Estimated and Actual Delay Improvement



Conclusions

- Propose novel restructuring algorithm
  - Achieves 11.7% delay improvements
  - Minimizes negative impact to other performance metrics
  - Performs efficiently on larger benchmarks
- Introduce a non-monotonicity metric to identify important parts of the circuit for optimization
- Develop a simulation-based abstraction for performing synthesis
  - Quickly evaluate the feasibility of several topologies
  - Exploit global don't cares to enhance optimization potential