Safe Delay Optimization for Physical Synthesis

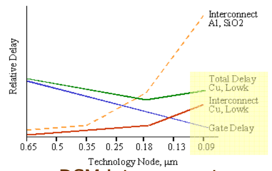
Kai-hui Chang,
Igor L. Markov and Valeria Bertacco

University of Michigan at Ann Arbor

Jan. 25, 2007

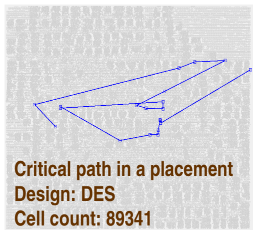
Improving Deep Submicron Layouts

- Current technology trends ($\leq$ 90nm)
  - Interconnect delay is dominant
  - Impact of logic restructuring is difficult to predict
  - Use of placement information is critical



DSM interconnect
(sources: TSMC/IBM)

- Existing post-placement timing optimizations break down
  - Logic level timing analysis is inaccurate (route length and net delay estimates are arbitrary)
  - Estimated improvements may worsen timing
  - May increase congestion and route length
  - Critical nets may detour during routing
  - **Lack of predictability**



Our Work: Improving Predictability

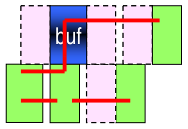
- We define a new parameter of physical synthesis optimizations: physical safeness
- We propose a new safe physical synthesis technique
  - Predictable delay improvement
  - Easily verifiable correctness
  - Up to 86% improvement for IWLS2005 benchmarks with < 1% increase in route length and via count
  - 11% delay improvement on average

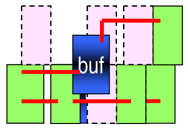
Outline

- Physical safeness
- Our physical synthesis approach
- Experimental results
- Conclusions

Physical Safeness

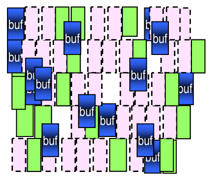
- Preserving physical parameters
  - Timing, congestions, distances, locations
- Safe techniques allow only legal changes
  - Accurate analysis can be performed
  - Changes that worsen layout are rejected immediately
- Unsafe techniques allow overlaps and route length increase
  - Legalization is required
  - Accurate analysis cannot be performed immediately





Physical Safeness

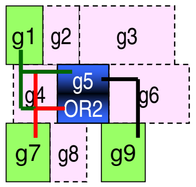
- Preserving physical parameters
  - Timing, congestions, distances, locations
- Safe techniques allow only legal changes
  - Accurate analysis can be performed
  - Changes that worsen layout are rejected immediately
- Unsafe techniques allow overlaps and route length increase
  - Legalization is required
  - Accurate analysis cannot be performed immediately



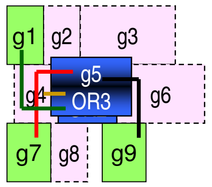
Batched legalization leads to unpredictability

Safe/Unsafe Examples: Rewiring

- Symmetry-based rewiring
- Physically safe



- ATPG-based rewiring
- Physically unsafe



Must call legalizer to remove overlaps

Physical Synthesis Techniques

Technique	Physical safeness	Optimization strength
Symmetry-based rewiring	Safe	Low
ATPG-based rewiring, buffer insertion, gate sizing, gate relocation	Unsafe	Low
Gate replication	Unsafe	Medium
Restructuring	Unsafe	High
Safe Resynthesis	**Safe**	**Medium**

Safe Resynthesis is useful by itself or after unsafe techniques for further optimization

Outline

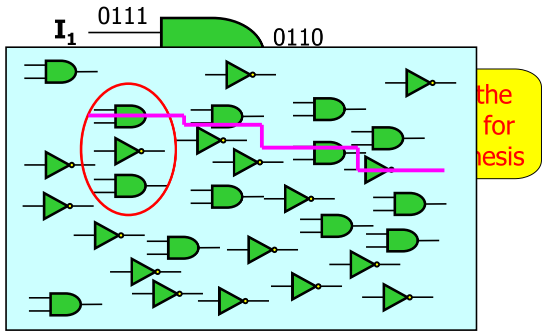
- Physical safeness
- Our physical synthesis approach
  - Naïve variant
  - Enhanced variant
- Experimental results
- Conclusions

Safe Resynthesis

1. Simulate patterns and generate a signature for each wire in the circuit

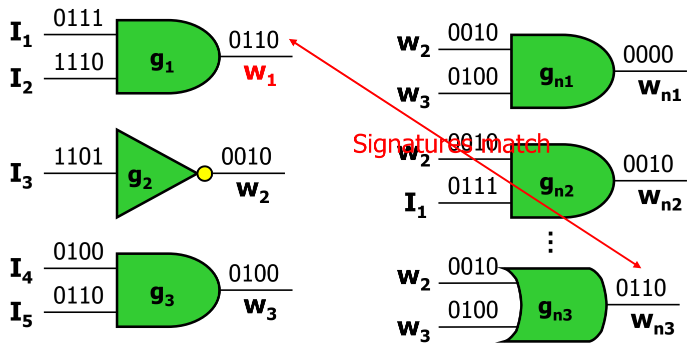
Input vectors

I_1	I_2	I_3	I_4	I_5
0	1	1	0	0
1	1	1	1	1
1	1	0	0	1
1	0	1	0	0



Safe Resynthesis (Naïve Approach)

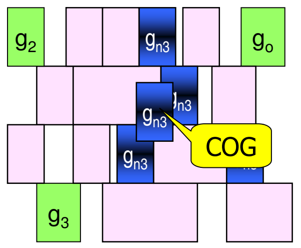
2. Resynthesize the target wire w_1 with combinations of different gates and wires



(naïve: may end up trying gates of all types at all locations)

Safe Resynthesis

3. Place the new gate at overlap-free sites near the center-of-gravity of its inputs and outputs



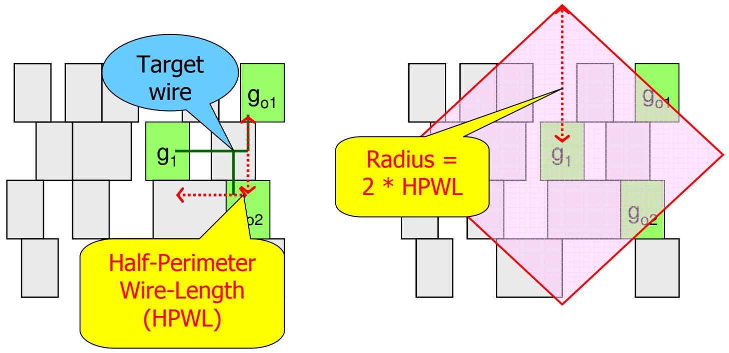
The location with the maximum improvement will be chosen
Equivalence checking verifies its correctness

Safe Resynthesis (Faster Approach)

- Trying all possible combinations is too expensive – will be improved
- Efficient search pruning
  1. Physical constraints: improves timing?
     (based on arrival times, locations and distances)
  2. Logical constraints: preserves functionality?
     (based on controlling values of gates)
- Resynthesis performed only if all constraints are satisfied

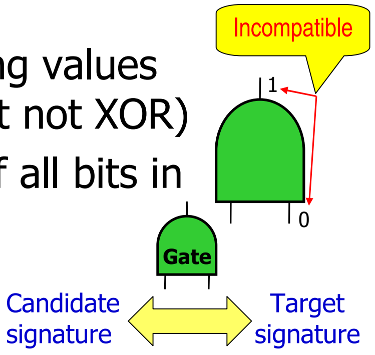
Pruning 1: Physical Constraints

1. Use arrival times (AT) from incremental STA
   - Consider only gates with $AT_{gate} < AT_{target}$
2. Try only gates close to the original driver



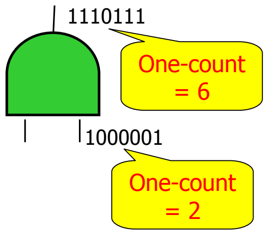
Pruning 2: Logical Compatibility

- Based on the controlling values of gates (AND, OR, but not XOR)
- Checks compatibility of all bits in
  - Candidate signature
  - Target signature
- Ignore wires with incompatible signatures
  - Reduce the number of candidate wires



Implementation Insights

- Accelerate compatibility test
  - "One-count": number of 1s in the signature
  - E.g., one-count decreases for an AND gate
- Improve signature quality
  - Poor signatures require more equivalence checking
  - Uses patterns produced by the FRAIG package in ABC synthesis package (UCB)
    - Distinguish different wires in an AIG



Analysis of Our Approach

- Scalability
  - Signature + equivalence checking scales better than BDDs in terms of memory usage
  - Can handle 100K gate designs
- Optimization power
  - Utilizes complete controllability don't-cares
  - Subsumes gate relocation and replication
  - Finds long range opportunities
- Safeness
  - Accurate analysis can be performed for each change

Outline

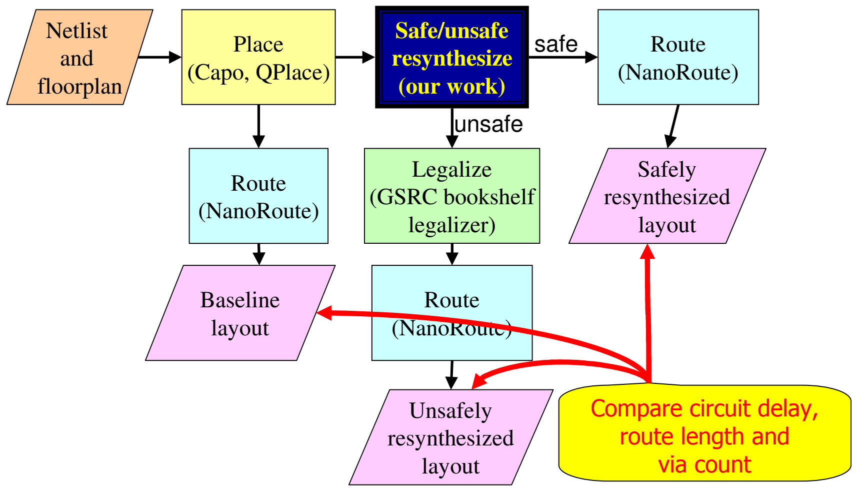
- Physical safeness
- Our physical synthesis approach
- Experimental results
- Conclusions

Experimental Setup

- Placer: Capo and QPlace
- Timing analyzer:
  - Before routing: D2M + Steiner tree
  - After routing: routed nets
- Benchmarks: IWLS2005, 0.18 μ m library

Suite	Benchmarks
OpenCores	SPI, DES_AREA, TV80, SYSTEMCAES, MEM_CTRL, AC97, USB, PCI, AES WB_CONMAX, Ethernet, DES_PERF
Faraday	DMA
ITC99	B14, B15, B17, B18, B22
ISCAS89	S35932, S38417

Our Experiments


Netlist and floorplan
Place (Capo, QPlace)
Safe/unsafe resynthesize (our work)
safe
Route (NanoRoute)
unsafe
Route (NanoRoute)
Legalize (GSRC bookshelf legalizer)
Safely resynthesized layout
Baseline layout
Route (NanoRoute)
Unsafely resynthesized layout
Compare circuit delay, route length and via count

Delay Improvement (30% Whitespace)

Bench mark	Estimated delay improvement				Routed delay improvement		
	Safe resynth.	Unsafe resynth.		Unsafe+ safe resynth.	Safe resynth.	Unsafe resynth.	Unsafe +safe resynth.
		Before legal.	After legal.				
AC97	2.67%	3.67%	3.44%	3.67%	1.56%	1.31%	2.65%
USB	5.21%	5.29%	5.10%	5.29%	3.09%	6.69%	10.41%
PCI	5.99%	5.37%	4.58%	5.37%	0.00%	-1.90%	0.00%
AES	2.32%	5.06%	4.94%	5.06%	2.25%	3.61%	5.66%
WB	61.37%	61.54%	61.48%	61.54%	61.29%	61.30%	63.14%
Ether.	85.66%	86.41%	85.89%	86.41%	85.61%	82.07%	86.60%
DES	1.98%	2.21%	2.12%	2.21%	1.93%	0.49%	2.44%
Ave.	23.60%	24.22%	23.93%	24.22%	22.25%	21.94%	24.41%

Improvement: unsafe+safe > safe > unsafe; unsafe resynthesis may worsen routed timing

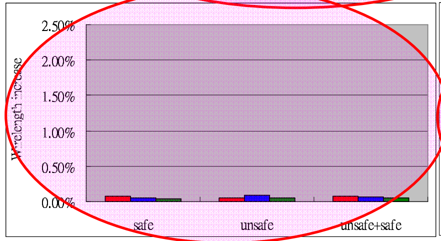
Delay Improvement (Different Percentage of Whitespace)

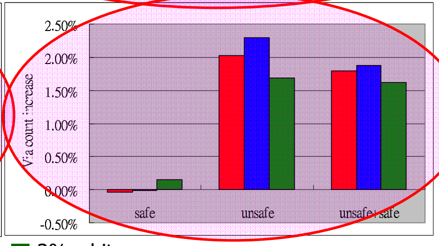
Percentage of whitespace	Estimated delay improvement				Routed delay improvement		
	Safe resynth.	Unsafe resynth.		Unsafe + safe resynth.	Safe resynth.	Unsafe resynth.	Unsafe +safe resynth.
		Before legal.	After legal.				
30%	23.60%	24.22%	23.93%	24.22%	22.25%	21.94%	24.41%
10%	23.59%	24.12%	23.64%	24.01%	23.52%	23.56%	23.98%
3%	20.33%	20.78%	20.34%	21.63%	20.22%	20.23%	21.38%

- **Safe and unsafe resynthesis have similar performance**
- **Unsafe+safe resynthesis achieves the most improvement**

Route Length and Via Count Increase (Different Percentage of Whitespace)

Percentage of whitespace	Route length increase			Via count increase		
	Safe resynth.	Unsafe resynth.	Unsafe+safe resynth.	Safe resynth.	Unsafe resynth.	Unsafe +safe resynth.
30%	0.08%	0.05%	0.08%	-0.04%	2.03%	1.80%
10%	0.05%	0.09%	0.07%	-0.01%	2.29%	1.87%
3%	0.04%	0.05%	0.05%	0.15%	1.68%	1.62%





🟥 30%, 🟦 10%, 🟩 3% whitespace

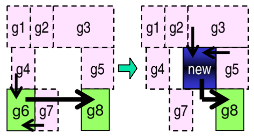
Route length increase is small for all layouts, while via count increase is significant for layouts produced by unsafe resynthesis

Summary of Empirical Results

- 22% smaller delay at 30% whitespace
- 20% smaller delay at 3% whitespace
- Route length and via count increase < 1%
- Unsafe optimization
  - Provides better improvement before legalization and routing
  - Improvement after routing is hard to predict
  - Increases via count
- Safe optimization
  - Effects are more predictable
  - Does not increase via count

Conclusions

- Physical safeness
  - Effects of unsafe techniques are hard to evaluate
  - Safe techniques may provide better improvement
- A safe resynthesis technique
  - Up to 86% delay improvement
  - Route length and via count increase by less than 1%



- Unsafe + safe optimization leads to the most delay improvement
  - More powerful safe optimizations
  - Techniques to apply unsafe optimizations in a safe way