Fast Synthesis

DC Explorer Perspective

Jovanka Ciric Vujkovic, Ph.D.
Senior Manager R&D, Design Group, Synopsys Inc.

Workshop on Logic Synthesis and Verification
December 10th, 2015
Lausanne, Switzerland
Outline

Fast Synthesis in Today’s Designs

Fast Synthesis in DC Explorer

Synthesis Flow With Fast Gate Sizing

Conclusion and Future Work
Fast Synthesis in Today’s Designs
Growing Complexity Requires Rethinking of Design Strategy

Today’s Challenges

• In North America, about 20% of the designs exceed 100M gates

• Higher clock speed trend

• Multiple voltage domains

• Complex SDC constraints

• Large number of IPs

• Schedules of 15 months or less

---

Source: Synopsys’ Global User 2015 Survey
Growing Design Complexity Requires Fast and Early Exploration

- During early design stages
  - RTL and constraints are incomplete
  - Many blocks and 3rd-party IPs are incomplete or unavailable
  - Floorplan is unavailable or preliminary

- Need an efficient way to:
  - Resolve data inconsistencies
  - Debug timing constraints
  - Improve design data

- Fast synthesis requires high-quality netlists and reduces design schedules
Fast Synthesis in DC Explorer
DC Explorer in Design Cycle

Better Starting Point For RTL Synthesis

• Tolerance for incomplete data
  – Faster RTL and constraints development
  – Pre-Synthesis floorplanning

• 5-10X faster runtime compared to final RTL synthesis
  – Quick what-if analyses

• Physical implementation
  – Reading floorplans
  – Congestion-driven placement
  – Physically aware optimizations

• 8% timing and area correlations
  – Early visibility into synthesis results
DC Explorer
*Early Design Exploration*

- Up to one month faster schedule
- Early visibility
  - Tolerance for incomplete data
  - Low-power support
  - Floorplan exploration
- Debug and RTL cross-probing
  - Timing analysis
  - Logic-level analysis
  - Congestion analysis
5-10X Faster Runtime Compared to Final RTL Synthesis

• New fast optimization technology
  – Unaffected by the quality of constraints
  – Multicore support delivers additional 2X faster runtime on 4 cores

• Enabling rapid what-if explorations of design configurations
8% Correlation with Design Compiler

• Assess the likelihood of meeting design targets
  • Support power and test
    – Clock gating, %LVT leakage optimizations, scan insertion, and test DRC checks
  • Identify potential improvements before implementation
    – Datapath architecture

Timing Correlation

Area Correlation
How to Design a Fast Synthesis Flow

• Principles for achieving faster runtime
  – Re-think old and devise new faster algorithms
  – Create a convergent flow
  – Approximate only when QoR impact is minimal
  – Reduce effort for iterative algorithms
  – Exploit design characteristics

• Have state-of-the-art Design Compiler
  Graphical reference flow
  – Important to achieve tight correlation with final synthesis
  – No missing functionalities
No Compromise on Features

Fast synthesis in DC Explorer supports all major optimizations and engines

- Combinational Optimizations for timing and area
  - Boundary optimization
  - Constant propagation
  - Datapath extraction and optimization
- Sequential optimizations
  - Sequential output inversion
  - Unloaded and constant register removal
  - Register merging
  - Retiming
- Clock gating
- Combinational and sequential gate sizing
- High-fanout buffering
- Congestion-driven placement
- Multicorner multimode
- Multibit register mapping
Synthesis Flow with Fast Gate Sizing
Gate Sizing

- Gate sizing optimization assigns gate sizes to all cells in the design using the technology library model for each gate to meet timing constraints with minimal area and power.

- Prior work in gate sizing:
  - Continuous methods
    - Convex nonlinear optimization (numerical formulation, Lagrangian relaxation)
    - Linear programming and network flow
    - Slew budgeting
  - Discrete methods
    - Sensitivity-based iterative methods
    - Dynamic programming
    - Branch and bound

- Gate sizing algorithm in DC Explorer is based on numerical synthesis.
Patented Technology for Gate Sizing

**Abstract**

Techniques and systems are described for improving the efficiency of timing calculations in numerical sequential cell sizing and for improving the efficiency of incremental slack margin propagation. Some embodiments cache timing-related information associated with a source driver that drives an input of a sequential cell that is being sized, and/or timing-related information for each output of the sequential cell that is being sized. The cached timing-related information for the source driver can be reused when sizing a different sequential cell. The cached timing-related information for the outputs of the sequential cell can be reused when evaluating alternatives for replacing the sequential cell. Some embodiments incrementally propagate slack margins in a lazy fashion (i.e., only when it is necessary to do so for correctness or accuracy reasons) while sizing gates in the circuit design in a reverse-levelized processing order.

15 Claims, 3 Drawing Sheets
Numerical Synthesis

• Constraint-invariant synthesis
  – Linear runtime in the size of the design
  – Works on entire design instead of few critical paths

• Numerical synthesis
  – Advanced size-independent library modeling enables numerical formulation
  – Optimal solution using state-of-the-art numerical solvers
  – Works for sequential and combinational logic

Numerical Formulation = f(library cell timing, library pin cap, path stages, endpoint loading, startpoint cap, and so on)
Numerical Delay Modeling Basics

- Theory of Logical Effort by Sutherland et al. 1999
  - $g$: logical effort
  - $h$: electrical effort
  - $p$: parasitic delay of gate

- Can be rewritten as:

- Derivation of $g$ and $p$ for a library cell:
  - Not perfectly linear
  - Different delays for rise and fall times
  - Variance between different timing arcs
  - Slope variance for different input transitions

- Library analysis requires clustering and handling outliers for $g$ and $p$ derivation
Gate Sizing Algorithm

Iyer et al.

(21) Appl. No.: 13/954,523
(22) Filed: Jul. 30, 2013
(51) Int. Cl. G06F 17/50 (2006.01)
(52) U.S. Cl. 716/70. 716/88. 716/100
(58) Field of Classification Search
CPC: G06F 17/50; G06F 17/50
USPC: 716/88
See application file for complete search history.

15 Claims, 4 Drawing Sheets

ROBUST NUMERICAL OPTIMIZATION FOR OPTIMIZING DELAY, AREA, AND LEAKAGE POWER

Applicant: Synopsys, Inc., Mountain View, CA (US)
Inventors: Mahesh A. Iyer, Fernment, CA (US); Amir H. Mottaz, Los Alamos, CA (US)
Assignee: Synopsys, Inc., Mountain View, CA (US)

Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 14 days.

ABSTRACT

Systems and techniques are described for performing numerical delay, area, and leakage power optimization on a circuit design. During operation, an embodiment can iteratively perform the following set of operations in a loop, wherein in each iteration a current threshold voltage value is progressively decreased: (a) perform numerical delay optimization on the circuit design using a numerical delay model that is generated using gates in a technology library whose threshold voltages are equal to the current threshold voltage; (b) perform a total negative slack based buffer optimization on the circuit design; and (c) perform a worst negative slack stack optimization on the circuit design. The embodiment can then perform multiple iterations of worst negative slack stack optimization.

15 Claims, 4 Drawing Sheets

Receive a set discretized delay models, wherein each discretized delay model corresponds to a time of a cell of the cell type, and wherein each discretized delay model is capable of being represented by a set of points in a multi-dimensional space, the multi-dimensional space including an output loading dimension, an input transition dimension, an output delay dimension, and an output transition dimension.

Determine a set of specific numerical delay models based on the set of discretized delay models.

Determine the generic numerical delay model for the cell type based on at least a subset of specific numerical delay models.
Summary and Future Work
Summary and Future Work

• Designing fast synthesis with good QoR and must-have optimizations for today’s large designs is complex

• Tight correlation with final synthesis is a must

• Fast runtime in synthesis continues to be a major objective

• Modeling additional physical effects in smaller geometries to maintain correlation with increased design complexities

• New technologies are developed to speed up synthesis without QoR degradation, such as area, timing, and power
Thank You