Design Specific Circuit Element Selection

David Hathaway, Arun Venkataraman, John Cohn

IBM
Overview

- Objective
- Background
- Past design flow
- Current design flow
- Proposed design flow
- Open problems
- Conclusions
Objective

- Given a set of clone versions, and a set of constraints like delay slacks and a range of feasible sizes,
  - find the **minimum** number of new clone versions such that each clone version covers a maximal set of clone versions having similar constraints.
Chip Design Idea

3D Graphics Accelerator

High-level language description

for i := 1 to 32
add [i] <= a [i] + b [i]

high level synthesis and intermediate transformations like logic synthesis

optimized network of logic elements called gates

Logic Design Process
Library of Gates

- Several versions of the same gate
  - referred to henceforth as "clone" versions
- Each version needs extensive qualification to extract the following characteristics
  - delay rule
  - input capacitance
  - drive capability
  - power consumption
- Problems with having myriad clone versions
  - High cost of implementing clones
  - Significant data explosion
  - Increased qualification overheads
Library of gates

Optimization constrained by performance, area objectives and by the target library of gates
Why do we need "clones"?

- Why do we need "clones"?
- higher drive for less delay
- wider devices, more current
- increased silicon area
- larger input capacitance
- increased delay on prior stages
- increased power consumption

Diagram:

- AND3
- OR2
- NAND1
- NOR4
- INV1
- C_in
- F1
- F2
Original Design Flow

Original design

Unoptimized Boolean Network

Timing Analysis driven, heuristic gate assignment (using logic and physical transformations)

Analyzer(s)

Target library of logic elements

Optimized network consisting of "clone" versions present in the target library

Network might not be optimal with respect to performance or other secondary objectives
1-Dimensional Gate Sizing

\[ \text{NAND}_2 \]

\[ \text{NAND}_4 \]

resizing transistors for improving gate performance
Current Design Flow

Original design

Unoptimized Boolean Network

"continuous" one-dimensional gate size based tuning

Optimized network of "clone" versions not always corresponding to elements in target library

modify design

No

Target library of logic elements

Network comprising of clone versions present in the target library; (Network may no longer be optimal from the standpoint of performance or other secondary objectives)

have design objective(s) been met?

Yes

success

modify design

re-map network
Multidimensional Gate Sizing

minimize the slew of input B for a particular usage of the NAND gate
Proposed Design Flow

Original design

Unoptimized Boolean Network

"continuous" multidimensional gate size based tuning

Optimized network of "clone" versions (worst case each clone is unique)

modify design

have design objective(s) been met?

re-tune using "group" versions as objective (optional)

implement clone versions

Group clone versions so as to maximally cover network using minimal number of clones subject to constraints imposed by design objectives

Have design objective(s) been met?

No

Yes

HOW ??
**Constraint-based design tuning**

**Typical process**
- Use general purpose non-linear optimizer
- Constraints ensure timing is OK
- Other constraints possible
- Objective: minimize area

**Constraints**
- \( AT_Y \geq AT_A + d_{AY} (C_Y, S_A, Tx_Y) \)
- \( S_Y \geq f_{S_{AY}} (C_Y, S_A, Tx_Y) \)
- \( AT_Y \geq AT_B + d_{BY} (C_Y, S_B, Tx_Y) \)
- \( S_Y \geq f_{S_{BY}} (C_Y, S_B, Tx_Y) \)
- \( AT_X \geq AT_Y + d_{YX} (C_X, S_Y, Tx_X) \)
- \( S_X \geq f_{S_{YX}} (C_X, S_Y, Tx_X) \)
- \( AT_X \geq AT_C + d_{CX} (C_X, S_C, Tx_X) \)
- \( S_X \geq f_{S_{CX}} (C_X, S_C, Tx_X) \)
- \( AT_Z \geq AT_C + d_{CZ} (C_Z, S_C, Tx_Z) \)
- \( S_Z \geq f_{S_{CZ}} (C_Z, S_C, Tx_Z) \)
- \( AT_X \leq RAT_X \)
- \( AT_Z \leq RAT_Z \)

- Area = \( \sum Tx_i \)

\( AT = \) arrival time
\( D = \) delay
\( S = \) slew
\( C = \) capacitive load
\( Tx = \) transistor sizes
\( RAT = \) required arrival time
Constraint-based design tuning

Feasible region

Lowest feasible cost

Lowest cost
Recap

- Original design flows
  - Optimization using logic and/or physical transformations constrained by a predefined target library
    - One time cost associated with designing library elements
    - Negligible impact on design data volumes
  - Overconstrained by the target library; hence difficult to find optimal solution(s)

- Current design flows
  - Continuous one-dimensional gate sizing till objectives met; remap network using target library elements
    - No additional library design and qualification overheads
    - Negligible impact on design data volumes
  - Sub-optimal results as a consequence of the constraint imposed by one-dimensional nature of the problem
Open Problems

- Proposed flow
  - Continuous, Multidimensional gate sizing till objectives met
  - Re-group clones so as to maximally cover network with minimal violations of feasibility constraints obtained from tuner
  - Re-tune using re-grouped clones as objectives to the tuner (relaxation) - Optional but useful

- Claim(s)
  - Will provide faster design objective convergence with minimal loss of optimality with respect to secondary objectives
  - If re-grouping of clones efficient and effective, implementation overheads will be manageable
Problem Definition - 1

- Given a set of clone versions, find the \textbf{minimum} number of new clone versions such that each clone version covers a maximal set of clone versions

\begin{itemize}
\item \textbf{subject to} minimizing some measure of perturbation of overall design
\end{itemize}

Note: 2-D space is for CMOS Inverter; Each transistor in design adds a dimension to the solution space
Simple Example

Two gates, one parameter each

Original problem space

Clustering space

Reduced problem space
Solution to Problem-1 can violate feasibility

Feasible region

New clone version causes constraint violation

Lowest feasible cost

Lowest cost
**Problem Definition - 2**

- Given a set of clone versions, find the minimum number of new clone versions such that each clone version covers a maximal set of clone versions
  - subject to minimizing some measure of perturbation of overall design given feasibility direction constraints

Note: Clones that do not belong to the original cluster as a result of the feasibility direction constraint violation are treated as new clone versions for this problem
Simple Example

Include tight constraints from optimization

Original problem space

Clustering space

Reduced problem space
Problem Definition - 3

- Given a set of clone versions, find the minimum number of new clone versions such that each clone version covers a maximal set of clone versions
  - subject to minimizing some measure of perturbation of overall design given feasibility direction constraints with some limits on the range of values that are valid

Note the change in clustering as a consequence of using feasibility direction constraints and range of the values that are valid.
Simple Example
Include non-tight constraints from optimization

Original problem space

Clustering space

Reduced problem space
Problem Definition - 4

- Given a set of clone versions, find the minimum number of new clone versions such that each clone version covers a maximal set of clone versions.

  - subject to minimizing some measure of perturbation of overall design given feasibility direction constraints with some limits on the range of values that are valid AND including interactions between parameters.

If interactions between the elements with this cluster are violated by value taken by one of the parameters then we remove the violation to guarantee feasible optimality of the solution.
Conclusion(s)

- Proposed flow to obtain optimal designs hindered by not having a solution for the combinatorial optimization problem of re-grouping clones so as to maximally cover the network while minimally violating feasibility constraints

- Secondary problem of bounding the number of new clone versions created will be based on the results obtained by solving the primary problem