Homework Questions, January 24, 2007

Placement Lecture (Lecture #5)

- **Question 1. Slide #49 (Simulated Annealing)**
  - Explain what is wrong with the picture of the evolution of the cost function. Assume that the “T” (temperature) parameter in SA is monotonically decreasing from a large initial value down to zero.

- **Question 2. Slide #46 (Quadratic Placement)**
  - Quadratic placement methods implicitly assume that the netlist hypergraph is represented as a graph. (The matrix [A] in the slide is derived from the adjacency matrix of the graph.) Explain how one should represent a hypergraph using a graph, in a manner that is “fair” and “efficient”.

Courtesy K. Keutzer et al. UCB
VLSI Design Flow and Physical Design Stage

- **Definitions:**
- **Cell:** a circuit component to be placed on the chip area. In placement, the functionality of the component is ignored.
- **Net:** specifying a subset of terminals, to connect several cells.
- **Netlist:** a set of nets which contains the connectivity information of the circuit.
What Is The Router’s Purpose?

- Router ultimately responsible for meeting specs/assumptions
  - Slew, noise, delay, critical-area, antenna ratio, PSM-amenable ...
- Checks performability throughout top-down physical implementation
  - Actively understands, invokes analysis engines and macromodels
- Many functions
  - Circuit-level IP generation: clock, power, test, package substrate routing
  - Pin assignment and track ordering engines
  - “Monolithic” (entire net at a time) topology optimization engines
  - Owns key DOFs: small re-mapping, device-level
  - Is hierarchical, scalable, incremental, characterized (e.g., coarse/quick routing), ...

---

**Advanced routing for deep submicron technologies**

sizes, higher clock speeds, and lower supply voltages in 2001 will call for a new IC rout...
Routing Applications

Cell-based

Mixed Cell and Block

Block-based
Standard-Cell Routing Example 2
Standard-Cell Routing Example 3
Routing Phases

- **Global routing**
  - Guide the detailed router in large design
  - May perform quick initial detail routing
  - Commonly used in cell-based design, chip assembly, and datapath
  - Also used in floorplanning and placement

- **Detail routing**
  - Connect all pins in each net
  - Must understand most or all design rules
  - May use a compactor to optimize result
  - Necessary in all applications
Variable-Die vs. Fixed-Die

- **Variable-Die**
  - Traditional channel-based routing formulations
    - Heritage: early standard-cell methodology
  - Used when only 2 or 3 layers of metal
  - If the router gets stuck, expand the channel \( \rightarrow \) die size is not fixed a priori

- **Fixed-Die**
  - Modern area-based routing formulations
    - Heritage: gate-array methodology
  - Used with N-layer metal (\( N > 3 \))
  - Placement of cells and cell rows is fixed
  - If the router gets stuck, have to go back to placement or floorplanning stage
Taxonomy of VLSI Routers

- Global
  - Graph Search
  - Steiner
  - Iterative

- Detailed
  - Restricted
    - River
    - Switchbox
    - Channel
  - General Purpose
    - Maze
    - Line Probe
    - Line Expansion

- Specialized
  - Power & Ground
  - Clock

Hierarchical
Greedy
Left-Edge

Courtesy K. Keutzer et al. UCB
Global Routing

- Objectives
  - Minimize wire length
  - Balance congestion
  - Timing driven
  - Noise driven
  - Keep buses together

- Frameworks
  - Steiner trees
  - Channel-based routing
  - Maze routing
Global Routing Formulation

Given  
(i) Placement of blocks/cells  
(ii) channel capacities

Determine  
Routing topology of each net

Optimize  
(i) max # nets routed  
(ii) min routing area  
(iii) min total wirelength

Classic terminology: In general cell design or standard cell design, we are able to move blocks or cell rows, so we can guarantee connections of all the nets (“variable-die” + channel routers).

Classic terminology: In gate-array design, exceeding channel capacity is not allowed (“fixed-die” + area routers).

Since Tangent’s Tancell (~1986), and > 3LM processes, we use area routers for cell-based layout
Global Routing

- Provide guidance to detailed routing (why?)
- Objective function is application-dependent
Graph Models for Global Routing

- Global routing problem is a graph problem
- Model routing regions, their adjacencies and capacities as graph vertices, edges and weights
- Choice of model depends on algorithm
- Grid graph model
  - Grid graph represents layout as a $h \times w$ array, vertices are layout cells, edges capture cell adjacencies, zero-capacity edges represent blocked cells
- Channel intersection graph model for block-based design
Global Routing Approaches

- Can route nets:
  - Sequentially, e.g. one at a time
  - Concurrently, e.g. simultaneously all nets

- Sequential approaches
  - Sensitive to ordering
  - Usually sequenced by
    - Criticality
    - Length
    - Number of terminals

- Concurrent approaches
  - Computationally hard
  - Hierarchical methods used
Sequential Approaches

- Solve a single net routing problem
- Differ depending on whether net is two- or multi-terminal

**Two-terminal algorithms**
  - Maze routing algorithms
  - Line probing
  - Graph shortest-path based algorithms

**Multi-terminal algorithms**
  - Steiner tree algorithms
Two-Terminal Routing: Maze Routing

- Maze routing finds a path between source (s) and target (t) in a planar graph.
- Grid graph model is used to represent block placement.
- Available routing areas are unblocked vertices, obstacles are blocked vertices.
- Finds an optimal path.
- Time and space complexity $O(h \times w)$.
Maze Routing

- Point to point routing of nets
- Route from source to sink
- Basic idea = wave propagation (Lee, 1961)
  - Breadth-first search + back-tracing after finding shortest path
  - Guarantees to find the shortest path
- Objective = route all nets according to some cost function that minimizes congestion, route length, coupling, etc.
Maze Routing

- Initialize priority queue Q, source S and sink T
- Place S in Q
- Get lowest cost point X from Q, put neighbors of X in Q
- Repeat last step until lowest-cost point X is equal to the sink T
- Rip and reroute nets, i.e., select a number of nets based on a cost function (e.g., congestion of regions through which net travels), then remove the net and reroute it
- Main objective: reduce overflow
  - Edge overflow = 0 if num_nets less than or equal to the capacity
  - Edge overflow = num_nets – capacity if num_nets is greater than capacity
  - Overflow = Σ (edge overflows) over all edges

Courtesy K. Keutzer et al. UCB
Maze Routing Cost Function and Directed Search

- Points can be popped from queue according to a multivariable cost function
- Cost = function(overflow, coupling, wire length, …)
- Add <distance to sink> to cost function → directed search
  - Allows maze router to explore points around the direct path from source to sink first

S denotes the source point
T is the sink point

Directed search limits the search space when all other cost variables are equal

Non directed search expands in a circular fashion from S
Directed search expands in a conical fashion from S to T
Limiting the Search Region

- Since majority of nets are routed within the bounding box defined by S and T, can limit points searched by maze router to those within bounding box
  - Allows maze router to finish sooner with little or no negative impact on final routing cost
  - Router will not consider points that are unlikely to be on the route path

![Diagram](image)

- S denotes the source point
- T is the sink point
- Bounding box of S and T
- Normally, the search region is restricted to the bounding box + X
- In this example, X = 2
- The points outside of blue area are not considered by the maze router
Problems With Maze Routing

Slow: for each net, we have to search $N \times N$ grid
Memory: total layout grid needs to be kept $N \times N$

Improvements
- Simple speed-up
- Minimum detour algorithm (Hadlock, 1977)
- Fast maze algorithm (Soukup, 1978)
  - depth-first search until obstacle
  - breadth-first at obstacle
  - until target is reached
- Will find a path if it exists, may be suboptimal
- Typical speed-up 10-50x

Further improvements
- Maze routing infeasible for large chips
- Line search (Mikami & Tabuchi, 1968; Hightower, 1969)
- Pattern routing
Line-Probe Algorithm

*Mikami & Tabuchi* *IFIPS Proc, Vol H47, pp 1475-1478, 1968*

**Mikami + Tabuchi’s algorithm**
- Generate search lines from both source and target (level-0 lines)
- From every point on the level-i search lines, generate perpendicular level-(i+1) search lines
- Proceed until a search line from the source meets a search line from a target
- Will find the path if it exists, but not guaranteed to find the shortest path

Time and space complexity: $O(L)$, where $L$ is the number of line segments
Line-Probe Summary

- Fast, handles large nets / distances / designs
- Routing may be incomplete
Pattern-Based Routing

- Restrict routing of net to certain basic templates
- Basic templates are L-shaped (1 bend) or Z-shaped (2 bends) routes between a source and sink
- Templates allow fast routing of nets since only certain edges and points are considered
- Simultaneous selection of patterns for all nets: combinatorial optimization
- See, e.g., Li/Carothers (1996)
Connecting Multi-Terminal Nets

In general, maze and line-probe routing are not well-suited to multi-terminal nets

Several attempts made to extend to multi-terminal nets

- Connect one terminal at a time
- Use the entire connected subtrees as sources or targets during expansion
- Ripup/Reroute to improve solution quality (remove a segment and re-connect)

• Results are sub-optimal
• Inherit time and memory cost of maze and line-probe algorithms
Multi-terminal Nets: Different Routing Options

(a) Steiner Tree (14)
(b) Steiner Tree with Trunk (15)
(c) Minimum Spanning Tree (16)
(d) Chain (17)
(e) Complete Graph (42)

Cost is determined by routing model
Steiner Tree Based Algorithms

- Tree interconnecting a set of points (demand points, D) and some other (intermediate) points (Steiner points, S)
- If S is empty, Steiner Minimum Tree (SMT) equivalent to Minimum Spanning Tree (MST)
- Finding SMT is NP-complete; many good heuristics
  - SMT typically 88% of MST cost; best heuristics are within ½ % of optimal on average
- Underlying Grid Graph defined by intersection of horizontal and vertical lines through demand points (Hanan grid) → Rectilinear SMT and MST problems
- Can modify MST to approximate RMST, e.g., build MST and rectilinearize each edge
Minimum Spanning Tree (Prim’s construction)

Given a weighted graph
Find a spanning tree whose weight is minimum

Prim’s algorithm
start with an arbitrary node $s$
$T \leftarrow \{s\}$
while $T$ is not a spanning tree
\[
\begin{cases}
\text{find the closest pair } x \in V - T, y \in T \\
\text{add } (x, y) \text{ to } T
\end{cases}
\]
runs in $O(n^2)$ time
very simple to implement
always gives a tree of minimum cost
Applying Spanning and Steiner Tree Algorithms

- General cell/block design: channel intersection graphs

- Standard-cell or gate-array design: RSMT or RMST in geometry or grid-graph
Problems with Sequential Routing Algorithms

Net ordering

- Must route net by net, but difficult to determine best net ordering!
- Difficult to predict/avoid congestion

What can be done

- Use other routers
  - Channel/switchbox routers
  - Hierarchical routers
- Rip-up and reroute
Global Routing: Concurrent Approaches

- Can formulate routing problem as integer programming, solve simultaneously for all nets

**Given**

(i) Set of Steiner trees for each net  
(ii) Placement of blocks/cells  
(iii) Channel capacities

**Determine**

Select a Steiner tree for each net w/o violating channel capacities

**Optimize**

Min total wirelength

---

Courtesy K. Keutzer et al. UCB
Taxonomy of VLSI Routers

- **Routers**
  - **Global**
    - Graph Search
      - Steiner
    - Iterative
  - **Detailed**
    - Restricted
      - River
        - Switchbox
      - Channel
    - General Purpose
      - Maze
        - Line Probe
      - Line Expansion
  - **Specialized**
    - Power & Ground
      - Clock
    - Clock
Channel vs. Switchbox

- Channel may have exits at left and right sides, but exit positions are not fixed
- We may map exits to either lower or upper edge of a channel
- One dimensional problem
- Terminal positions on all four sides of a switchbox are fixed
- Two dimensional problem

Switchbox routing is more difficult
Channel Routing Problem

**Input:** Pins on the lower and upper edge

**Output:** Connection of each net

**Constraints (Assumption)**
1. grid structure
2. two routing layers. One for horizontal wires, the other for vertical wires
3. vias for connecting wires in two layers

**Minimize:**
1. # tracks (channel height)
2. total wire length
3. # vias
Basic Terminology:

- Fixed pin positions on top and bottom edges
- Classical channel: no nets leave channel
- Three-sided channel possible
Horizontal Constraint Graph (HCG)

1. Node $v_i$: represents a horizontal interval spanned by net $i$

2. There is an edge between $v_i$ and $v_j$ if horizontal intervals overlap

3. No two nets with a horizontal constraint may be assigned to the same track

4. Maximum clique of HCG establishes lower bound on # of tracks: $\#$ tracks $\geq$ size of maximum clique of HCG

Local density at column $C$, $ld(C) = \#$ nets split by column $C$

Channel Density $d = \max ld(C)$ over all $C$

Each net spans over an interval

Horizontal Constraint Graph (HCG) is an undirected graph with:
- vertex: net
- edge: $<n_j, n_k>$, if intervals $I_j, I_k$ intersect
Vertical Constraint Graph (VCG)

1. Node: represents a net

2. Edge (a1→a2) exists if at some column:
   - Net a1 has a terminal on the upper edge
   - Net a2 has a terminal on the lower edge
   - Edge a1→a2 means that Net a1 must be above Net a2

3. Establishes lower bound: \# tracks ≥ longest path in VCG

4. VCG may have a cycle!
Doglegs in Channel Routing

Doglegs may reduce the longest path in VCG

Doglegs break cycles in VCG

Courtesy K. Keutzer et al. UCB
Channel routing problem is completely characterized by the vertical constraint graph and the horizontal constraint graph.
Detailed Routing Objectives

- Routing completion
- Width and spacing rule
  - Minimum width and spacing
  - Variable width and spacing
    - Connection
    - Net
    - Class of nets
  - Tapering
Detailed Routing Objectives

- Width and spacing rule

![Width-based Spacing](image-url)

Minimum spacing

Width-based Spacing
Detailed Routing Objectives

- Via selection
  - Via array based on wire size or resistance
  - Rectangular via rotation and offset

Rotate and offset horizontal vias

No rotation for a “cross” via
Detailed Routing Objectives

- Understand complex pin & equivalent pin modeling
Detailed Routing Objectives

- Noise-driven

- Spacing
  - Extra space

- Segregation
  - Noisy region
  - Quiet region

- Shielding
  - Grounded Shields
Detailed Routing Objective

- Shielding
  - Same-layer shielding
  - Adjacent-layer shielding
Detailed Routing Objective

- Shielding
  - Bus shielding
  - Bus interleaving
Detailed Routing Objectives

- Differential pair routing
- Balanced length or capacitance

Differential pair routing

Balanced length or capacitance
Detailed Routing Objectives

- Bus Routing
Detailed Routing Objectives

- Process antenna rule
- Phase shift mask
- Other manufacturability objectives
Compaction

- **Channel Compaction (one dimension)**
Compaction

- **Area Compaction** (1.5 or 2 dimension)
  - May need a lot of constraints to get desired results
Shape-Based Routing

- Evolved from maze routing
- Gridless: look at actual size of each shape
- Each shape may have its spacing rule
- Good for designs with multiple width/spacing rules and other complex rules
- Slower than gridded router
Incremental Routing → “ECO”

- ECO == Engineering Change Order
  - Cf. Kahng/Mantik, Incremental Optimization
- Re-route with minor local adjustment
- Need rip-up and reroute capability
- Difficult to confine perturbation when compactor is used
Clock Routing

Balanced Tree

H-Tree
Clock Routing

- Multiple Clock Domains

Trunk or Grid

Clock Mesh
Power Routing

- Power Mesh
- Power Ring
- Star Routing
Modern Routing Rules (1)

- Minimum area rules and via stacking
  - Stacking vias through multiple layers can cause minimum area violations (alignment tolerances, etc.)
  - *Via cells* can be created that have more metal than minimum via overlap (used for intermediate layers in stacked vias)

- Multiple-cut vias
  - Use multiple-cut vias cells to increase yield and reliability
    - Can be required for wires of certain widths
  - Multiple via cut patterns have different spacing rules
    - Four cuts in quadrilateral; five cuts in cross; six cuts in 2x3 array; …
    - With wide-wire spacing rules, complicates pin access
  - Cut-to-cut spacing rules \(\Rightarrow\) check both cut-to-cut and metal-to-metal when considering via-to-via spacing

- Line-end extensions
  - Vias or line ends need additional metal overlap (0\textsuperscript{th}-order OPC)
Modern Routing Rules (2)

- **Width- and Length-dependent spacing rules**
  - Width-dependent rules: domino effects
  - Variant: “parallel-run rule” (longer parallel runs $\rightarrow$ more spacing)
  - Measuring length and width: *halo* rules affect computation

- **Influence rules or stub rules**
  - A fat wire, e.g., power/ground net, will influence the spacing rule within its surroundings $\rightarrow$ any wire that is $X$ um away from the fat wire needs to be at least $Y$ um away from any other geometry.
  - Example: fat wire with thin tributaries
    - bigger spacing around every wire within certain distance of the thin tributaries
    - ECO insertion of a tributary causes complications
    - Strange jogs and spreading when wires enter an influenced area
Example: LEF/DEF 5.5, April 2003

LEF/DEF 5.5 Language Reference
LEF Syntax

Either width > 1.50 μm and parallel length > 0.50 μm
Either width > 3.00 μm and parallel length > 3.00 μm
Either width > 5.00 μm and parallel length > 5.00 μm

0.50 μm spacing
1.00 μm spacing
2.00 μm spacing

Figure 1-7

These rules translate into the following SPACINGTABLE PARALLELRUNLENGTH statement:

```
LAYER m1
...
SPACINGTABLE
PARALLELRUNLENGTH 0.00 0.50 3.00 5.00 # lengths must be increasing
WIDTH 0.00 0.15 0.15 0.15 0.15 0.15 # max width > 0.00
```

Andrew B. Kahng, UCSD
Example: LEF/DEF 5.5, April 2003

Spacing rules require a minimum spacing value to ensure that nearby wires are separated. Even if the wires are narrow, Figure 1-8 on page 55 illustrates this situation. Use the following SPACINGTABLE INFLUENCE syntax to describe this table:

```
SPACINGTABLE INFLUENCE
   {WIDTH width WITHIN distance SPACING spacing} ...
```

If a wire has a width that is greater than `width`, and the distance between it and two other wires is less than `distance`, the other wires must be separated by spacing that is greater than or equal to `spacing`. Typically, the `distance` and `spacing` values are the same. Note that the distance halo extends horizontally, but not into the corners.

By definition, the width is the smaller dimension of the object (that is, the width is less than or equal to the length of the large wire).

Figure 1-8
Modern Routing Rules (3)

Density
- Grounded metal fills (dummy fill*)
- Via isodensity rules and via farm rules (via layers must be filled and slotted, have width-dependent spacing rule analogs, etc.)

Non-rectilinear ($\lambda$-geometry) routing
- X-Architecture:  http://www.xinitiative.org/
  - Y-Architecture:  http://vlsicad.ucsd.edu/Yarchitecture/ , LSI Logic patents
- Landing pad shapes (isothetic rectangle vs. octagon vs. circle), different spacings (~1.1x) between diagonal and Manhattan wires, etc.

More exceptions
- More non-default classes (timing, EM reliability, …)
  - Not just power and clock
- >0.25µm width may be “wide” → many exceptions
**Via Doubling For Redundancy**

- **Single-cut via** →
  - Double-cut via (“via doubling”)
    - During Routing?
    - **or** After Routing?

- **Why redundant vias?**
  - Improve yield: redundant via is “insurance” for via opens
  - Improve timing yield: the resistance of partially blocked vias increases
  - Reservoirs of metal for EM reliability
  - Etc.

- **Observation:**
  - The first 70-80% vias can be easily doubled even for very congested designs

**Can we improve more?**
Short-Loop Paths Also Add Redundancy

- Short-loop paths
  - Create detours to double vias which cannot otherwise be doubled with an adjacent single via
  - Up to 97.5% via doubling coverage is reported
  - May lead to antenna / timing issues

Can we have more?

→ 100% via doubling coverage ?!?
“Spread, Fatten, Fill …”

LPC hot spot

Hot spot removed
Problem: Charging and Antennas

- Process steps use plasmas, charged particles
  - Charge collects on conducting poly, metal surfaces
  - Capacitive coupling: large electrical fields over gate oxides cause damage or complete breakdown
    - Induced $V_t$ shifts affect device matching (e.g., for analog) and timing predictability

- Solution: limit \( \text{antenna ratio} = \frac{(A_{\text{poly}} + A_{M1} + \ldots)}{A_{\text{gate-ox}}} \)
  - \(A_{Mx}\) = metal\((x)\) area that is electrically connected to node without using metal \((x+1)\), **and** not connected to an active area
  - E.g., antenna ratio ~400 for thicker oxide, ~2000 for thin oxide
  - Several antenna rules: per-layer, cumulative, cap-based

- Bridging solution = break antenna by hopping to higher layer
  - *Extra wiring, vias, congestion*
  - Antenna ratios (and gate areas) decrease → more bridges

- Diode solution = Reverse-biased diode or source-drain contact near gate
  - *Leakage, area, timing penalties*
Problem: Resist Pattern Collapse

- $\Pr(\text{pattern collapse}) = f(\text{length})$
  - Length-dependent spacing rules
- Limits wire aspect ratio, packing density

- Futures: Standardized embedding of long wires for manufacturability and physical reliability

becomes

???

Cao et al. U. Wisconsin
Problem: Layout Density Control

- **Area fill**: electrically inactive; floating or grounded
- **Area fill insertion (and, slotting)**
  - Decreases local density variation
  - Decreases post-CMP ILD erosion, conductor dishing
Density Management and Futures

- Physical model based *density rules* and fill synthesis
  - Current “coevolved” state: Wrong rules, weak tools

[Image of circuit layout]
Density Management and Futures

- Performance Impact Limited Fill
  - E.g., Chen et al., DAC-03
  - Fill insertion driven by timing, coupling slack awareness
Density Management and Futures

- Splitting for uniformity
  - Slotting for uniform CMP replaced by splitting
  - Less data than traditional slotting
  - Power mesh: more accurate R/C analysis

- Combined density control and local pattern control (e.g., fill helps iso-dense, PSM phase-assignability)

- Details: hierarchy, reuse, multiple length scales, ...

Easy connections through standard via arrays

Difficult to connect - where should vias go?
Futures: Search in a Grid

- Routing today is composed from “optimal” source-target connections
- 35 years of enhancements
  - Best-first search (unidirectional and bidirectional)
  - Bit-packing, memory management
  - Connection ordering, costing, ripup-and-reroute
  - Parallel processing of overlapping switchboxes (search & repair)
- **Core weakness = Heuristic search framework**
  - E.g., costing, order-dependence, runtime and memory scaling
- **Gridded context is bad**
  - Expensive to model and push through wide connections
- **Fails when edge cost is path history-dependent**
  - E.g., stacked-via rules → cost of edge depends on the preceding edges in the connection → A* routing fails
Topological Routing

- “Pure” mechanism to avoid premature, arbitrary decisions
  - Topology and embedding are separated

- Conducive to via minimization and non-preferred direction routing frameworks

- Key performance-affecting attributes are easy to track
  - Identity of neighboring wire, parallel run length, bounds on spacing to neighbor, etc.
Non-Preferred Direction Routing – X Arch
Router Futures

- Sets of nets, not individual nets
- Topologies, not connections
- Patterns (= restricted layout solutions)
- “Analog” rules
Summary

- Many routing algorithms for various application contexts
- Maze routing algorithms and area-based (fixed-die) routing paradigm are dominant today
  - Difficult to handle complex requirements (crosstalk reduction, differential routing, restricted design rules, …)
  - Difficult to create “whole topologies” – esp. timing-driven
- Router futures
  - Sets of nets, not individual nets
  - Whole nets, not individual connections
  - Patterns = restricted layout solutions
  - “Analog” rules
- What makes routing harder in the future?
  - Manufacturing constraints → Via-doubling, spreading/tapering/filling, …
- What makes routing easier in the future?
  - Manufacturing constraints → One-pitch / one-orientation routing styles
  - Parallel / distributed processing
Question 1. Steiner Trees

The cost (wirelength) of a rectilinear Steiner minimum tree can be up to $\frac{1}{3}$ less than the cost of a rectilinear minimum spanning tree. Put another way, $\frac{c(\text{RMST})}{c(\text{RSMT})} \leq \frac{3}{2}$. Draw an example for which this so-called “performance ratio” is tight, i.e., the ratio is exactly $\frac{3}{2}$.

Question 2. Steiner Trees again

Find on the web at least 5 “good” software packages for solving the rectilinear Steiner minimum tree problem. List the URLs, along with commentary on each package’s respective generality or other merits. (At the minimum, please explain briefly why you believe a given package is likely to be “good”.)