Physical Design Flow

Input
- Read Netlist

Floorplanning
- Floorplanning
- Initial Placement

Placement
- Routing Region Definition
- Global Routing
- Cost Estimation
- Routing Region Ordering
- Detailed Routing
- Cost Estimation

Routing

Output
- Compaction/clean-up
- Write Layout Database

Placement Improvement
Routing Improvement
This Class + Logistics

- **Reading**
  - Partitioning, Placement, Routing, Compaction

- **Schedule**
  - Routing today
  - Perhaps back to Partitioning and Floorplanning in Lecture 10
  - Catch up on Lab component: STA, P&R
Homework Solutions

- HW #7: One type of answer is of form, “Add the node delay to all its incoming edges”, or “Split the node delay onto incoming and outgoing edges”. These are not ideal approaches, because some pin-to-pin path delays will be changed. The preferred solution is to create an explicitly pin-based timing graph, i.e., add an edge between every i-o pin pair of the gate, with edge delay equal to the gate delay. (For example, if a cell has 3 inputs and 2 outputs, it would induce 6 edges in the pin-based timing graph.) This will preserve all pin-to-pin path delays.

- HW #8: If a gate is resized, we will see a change in gate delay, gate drive strength, and gate input capacitance. This will affect all downstream path delays and arrival times (since propagated slews will change and thus propagated delays will change), and similarly will affect upstream delays and required times. It is necessary to recompute AATs or RATs, and slacks, in the union of combinational fanin and fanout cones of the resized gate. I.e., the union of all nodes on purely combinational upstream and downstream timing paths. There is a hitch in this answer: in a latch-based design, latch transparency may permit the effect of resizing to go beyond just the immediate fanout and fanin cones.

- HW #9: As the block is crunched down, wires become shorter and timing improves. Then, wires begin crowd each other, leading to congestion (possible detouring) and closest-neighbor coupling which degrades timing.
Homework Solutions

HW #10: Hold time (short path) violations can be fixed in a number of ways. The basic idea is to perform *padding* of short paths. Clock sink insertion delay (to the start of the short timing path) can be padded by adding buffers or snaking (serpentine routing), or dummy logic (loads). Path delay can be padded, again with buffers or snaking or dummy loads. Sizing of gates and interconnects can also increase delay without changing the circuit topology, by (i) increasing capacitive loads (gate upsizing) to slow down upstream drivers and gate transitions without recovering equivalent speed gains on driven stages, or (ii) increasing interconnect resistance (wire narrowing) to slow down wires. Finally, more powerful logic resyntheses, retiming, etc. could be applied but are generally discouraged because functional verification must then be redone. Making the system asynchronous, changing the clocking style, etc. are not really of any practical interest for an ASIC flow.

HW #11: See the 1997 D. Dobberpuhl paper. To reduce from 26W to 0.5W was approximately 13X from Vdd, process, frequency scaling (5.3X from reducing Vdd from 3.45V \(\rightarrow\) 1.5V; 2X from scaling Leff from 0.5um \(\rightarrow\) 0.25um; 1.25X from scaling frequency from 200MHz \(\rightarrow\) 160MHz) and 4X from other factors (multiple clocks plus edge-triggered flip-flops 1.3X; less functionality 3X).
Routing Applications

Cell-based

Mixed
Cell and Block

Block-based
Standard Cell Layout
Routing Algorithms

- **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

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

- Hierarchical
- Greedy
- Left-Edge
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
Channel Intersection Graph

- Edges are channels, vertices are channel intersections (CI), v1 and v2 are adjacent if there exists a channel between (CI₁ and CI₂). Graph can be extended to include pins.
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
    - 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
- 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 = \( \sum \) (edge overflows) over all edges
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

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
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

Courtesy K. Keutzer et al. UCB

Kahng & Cichy, UCSD ©2003
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
  - find the closest pair \( x \in V-T, y \in T \)
  - add \((x, y)\) to \( T \)

- 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
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

X \{ gridded, gridless\}
One Layer Routing: General River-Routing

- For clock, power, ground still may need to solve single-layer routing
- Two possible paths per net along boundary
  - Path = alternating sequence of horizontal and vertical segments connecting two terminals of a net
- Consider starting terminals and ending terminals
- Assume every path counter-clockwise around boundary
One Layer Routing: General River-Routing

- Create circular list of all terminals ordered counterclockwise according to position on boundary
One Layer Routing: General River-Routing

- Boundary-packed solution
- Flip corners to minimize wire length
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
- Hierarchical
- Greedy
- Left-Edge

Courtesy K. Keutzer et al. UCB
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)
(i) grid structure
(ii) two routing layers. One for horizontal wires, the other for vertical wires
(iii) vias for connecting wires in two layers

Minimize:
(i) # tracks (channel height)
(ii) total wire length
(iii) # vias
Channel Routing

- Basic Terminology:
  - Fixed pin positions on top and bottom edges
  - Classical channel: no nets leave channel
  - Three-sided channel possible

Courtesy K. Keutzer et al. UCB
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: $\# \text{ tracks} \geq \text{size of maximum clique of HCG}$

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

Channel Density $d = \max \text{ ld}(C)$ 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

Courtesy K. Keutzer et al. UCB
Vertical Constraint Graph (VCG)

1. Node: represents a net

2. Edge \((a_1 \rightarrow a_2)\) exists if at some column:
   - Net \(a_1\) has a terminal on the upper edge
   - Net \(a_2\) has a terminal on the lower edge
   - Edge \(a_1 \rightarrow a_2\) means that Net \(a_1\) must be above Net \(a_2\)

3. Establishes lower bound: # tracks \(\geq\) 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
Characterizing the Channel Routing Problem

Channel routing problem is completely characterized by the vertical constraint graph and the horizontal constraint graph.

Courtesy K. Keutzer et al. UCB
**Theorem**  A set of intervals with density \( d \) can be packed into \( d \) tracks.

Proof: \( I_1 = (a,b) \quad I_2 = (c,d) \)

Define: \( I_1 < I_2 \) iff \( b < c \) or \( I_1 = I_2 \)

1. reflexive: \( I_1 < I_1 \)
2. anti-symmetric: \( I_1 < I_2, I_2 < I_1 \rightarrow I_1 = I_2 \)
3. transitive: \( I_1 < I_2, I_2 < I_3 \rightarrow I_1 < I_3 \)

Set of intervals with binary relation \(<\) forms a partially ordered set (POSET)

Intervals in a single track \( \rightarrow \) form a chain
Intervals intersecting a common column \( \rightarrow \) form an antichain

Dilworth’s theorem (1950): If the maximum antichain of a POSET is of size \( d \), then the POSET can be partitioned into \( d \) chains
Left-Edge Algorithm for Interval Packing

Repeat
  create a new track t
  Repeat
    put leftmost feasible interval to t
  until no more feasible interval
until no more interval

Intervals are sorted according to their left endpoints

O(nlogn) time algorithm. Greedy algorithm works!
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**

![Diagram showing width and spacing rules](image)

**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

![Diagram showing simple and complex pins with strength levels: Strong, Weak, Must.](image-url)
Detailed Routing Objectives

• Noise-driven

Noisy region

Extra space

Grounded Shields

Quiet region

Spacing

Shielding

Segregation
Detailed Routing Objective

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

- Shielding
  - Bus shielding
  - Bus interleaving

Bus Shielding
Detailed Routing Objectives

- 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

![Diagram showing area compaction](image)
Shape-based Routing

- Evolve 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

- 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

Star Routing
Summary

- Various routing algorithms for different applications
- Maze routing algorithms and derivatives are okay for handling complex requirements
- Growing chip capacity and ever-changing process technology are major challenges to the router
Routing in Area Based Place-and-Route Tools

- Cadence Silicon Ensemble, for instance

- Clock and signal routing
  - Global route
  - Final route
  - Search and Repair
  - Final route optimization

- Gridded router → stays on grid, as close to pin as possible

- Clocks usually routed same way as signals, but earlier with other critical nets

- Violations before Search and Repair natural phenomenon
Routing Grids

- Routing grids
  - Detailed routing grid is grid for actual wire locations
  - Global routing grid is coarser grid for global routing
Routing Grids

Routing grids

- Detailed routing grid is distance between center lines of wires
- Routing grid is driven by technology design rules (spacing/width)
- The standard cells are designed for specific detailed routing grid
  - Cells need to be N multiples of detailed routing grid wide
  - Cells need to be M multiples of detailed routing grid tall

- Optimum grid:
  
  Smallest that is DRC correct
  Minimum distance between all metal
  
  Largest that does not waste space
  Minimum distance between metal in via cell

![Routing Grid Diagram]
Routing Grids

- Most common routing grid definition allows vias to be dropped diagonally
Final Routing

- All nets are always final routed

- Key philosophy
  - Unconnected net is a global problem → more difficult to fix
  - Net with violations is a local problem → less difficult to fix

- N by M Gcells are final routed at a time

- Nets are routed in preferred routing direction

- Final route creates violations at interfaces between Gcells
  - Caused by conflicting track assignments
  - Violations after final route are inherent to routing algorithm
  - Search and Repair is required to clean design
Search and Repair, Final Route Optimization

- Search and Repair: center new Gcells around violations

- If many Search and Repair passes are required, timing can degrade

- Option to redo global route for nets with violations
  - Potential timing degradation

- Final route optimization (after design is violation-clean)
  - Reduce wire length
  - Reduce via count
  - can potentially improve timing in processes with high via resistance
Process Antennas

- During manufacturing, static charge builds up on metal traces
  - Metal with static charge connected to gate $\rightarrow$ gate may be destroyed in electrostatic discharge $\rightarrow$ lower manufacturing yield
  - Antenna Ratio = maximum allowable ratio of metal area to gate area

- Solution 1: “Bridging” or “layer-hopping” to limit amount of metal connected to gate without safe discharge path (through reversed biased diode at output stage of logic gate) $\rightarrow$ more writing, vias, congestion

- Solution 2: Drop reverse-biased diode or source-drain contact close to gate to allow charge to escape harmlessly $\rightarrow$ area, power penalty

- Antenna Ratio worsening (decreasing) with successive technology nodes? E.g., 2000 $\rightarrow$ 400 $\rightarrow$ 100?

- Unfixed antenna yield penalty trades off against fixed antenna yield loss (from increased die size)

- Several antenna rules coexist: per-layer, cumulative, and new capacitance-based (parallel neighbors contribute to area)
Global and Local Congestion

- Globally over-congested design is too small

- Locally over-congested design may reflect
  - Bad I/O pin placement
  - Poorly placed or oriented blocks
  - Bad LEFs with inaccessible pins
  - Groups of cells with poor porosity (choose more porous cells or place groups of cell further apart)

- Congestion affects runtime, timing
  - Many Search and Repair passes will take a long time, and eventually degrade timing
Modern (Manufacturability) Considerations

- Minimum area rules and stacking vias
  - Stacking vias through multiple layers might 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
    - Can be required for wires of certain widths
  - But: multiple via cut patterns have different spacing rules
    - Four cuts in quadrilateral; five cuts in cross; six cuts in 2x3 array; …
    - along with wide-wire spacing rules, makes pin access DIFFICULT
  - Cut-to-cut spacing rules 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 in some technologies
More Modern Considerations

- **Width- and Length-dependent spacing rules**
  - Width-dependent rules: domino effects
  - Variant: “parallel-run rule” (longer parallel runs → more spacing)
  - How to measure 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 → 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

- **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.)
More Modern Considerations

- X-routing
  - [www.xinitiative.org](http://www.xinitiative.org)
  - Landing pad shapes (isothetic rectangle vs. octagon vs. circle), different spacings (~1.1x) between diagonal and Manhattan wires

- More exceptions
  - More non-default classes (timing, EM reliability, …) where before only power, clock had non-default rules
  - Even 0.3um wide wire is “fat” → many exceptions triggered
    - BTW, fat wires cause notch-filling problems

- Impact on routers
  - Before, issues were more benign and treatable by postprocessing
  - Now, routers need to natively consider these issues (How?)
  - E.g., stacked-via rules → cost of an edge depends on the preceding edges in the connection → A* maze routing fails