

ESE535: Electronic Design Automation

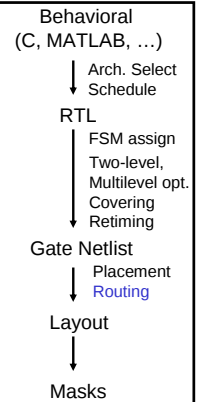
Day 21: April 15, 2009
Routing 1



Penn ESE535 Spring 2009 -- DeHon

Today

- Routing Cases
- Decomposition
- Channel Routing
- Variations
 - Over-the-cell



Penn ESE535 Spring 2009 -- DeHon

2

Routing Problem

- Where to wires run?
- Once know where blocks live (placement),
 - How do we connect them up?
 - *i.e.* where do the wires go?
 - In such a way as to:
 - Fit in fixed resources
 - Minimize resource requirements
 - (channel width → area)

Penn ESE535 Spring 2009 -- DeHon

3

Routing Cases

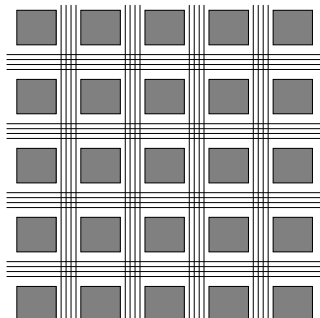
Gate-Array
Standard-Cell
Full Custom

Penn ESE535 Spring 2009 -- DeHon

4

Gate Array

- Fixed Grid
- Fixed row and column width
- Must fit into prefab channel capacity

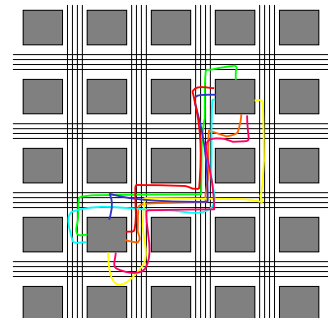


Penn ESE535 Spring 2009 -- DeHon

5

Gate Array

- Opportunities
 - Choice in paths
- How exploit freedom to:
 - Meet channel limits
 - Minimize channel width

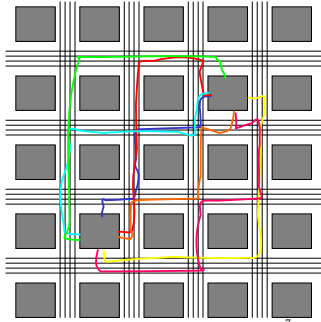


Penn ESE535 Spring 2009 -- DeHon

6

Gate Array

- Opportunities
 - Choice in paths
- How exploit freedom to:
 - Meet channel limits
 - Minimize channel width

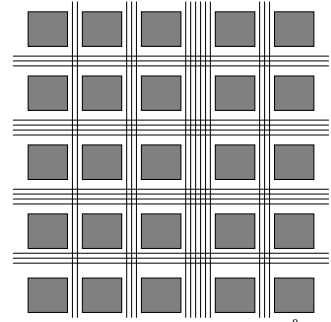


Penn ESE535 Spring 2009 -- DeHon

7

Semicustom Array

- Float Channel widths as needed
- Becomes a questions of minimizing total channel widths

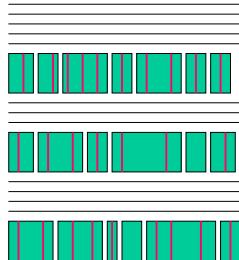


Penn ESE535 Spring 2009 -- DeHon

8

Row-based Standard Cell

- Variable size
 - Cells
 - Channels
- Primary route within row
- Vertical feed throughs

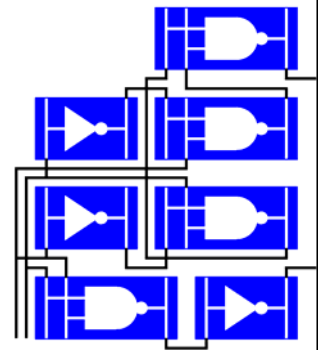


Penn ESE535 Spring 2009 -- DeHon

9

Standard Cell Gates

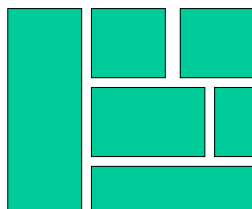
- IOs on one or both sides
- Design in Feed-thru



Penn ESE535 Spring 2009 -- DeHon

Full Custom / Macroblock

- Allow arbitrary geometry
 - Place larger cells
 - E.g. memory
 - Datapath blocks



Penn ESE535 Spring 2009 -- DeHon

11

Routing Decomposed

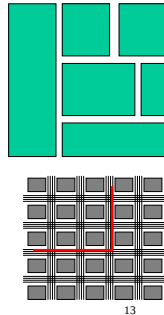
Penn ESE535 Spring 2009 -- DeHon

12

Phased Routing

After placement...

1. Slice (macroblock case)
 - And order channels
2. Global Route
 - Which channels to use
 - (suitable approach next class)
3. Channel Route
 - Today
4. Switchbox Route

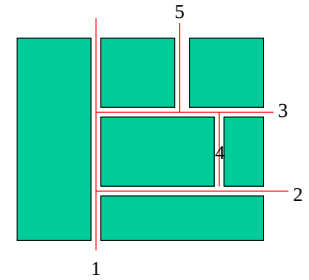


Penn ESE535 Spring 2009 -- DeHon

13

Macroblock → Channel Route

- **Slice** into pieces
- Route each as channel
- Work inside out
- Expand channels as needed
- Complete in one pass

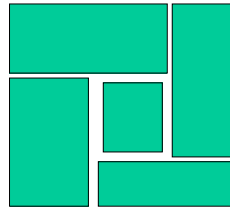


Penn ESE535 Spring 2009 -- DeHon

14

Not all Assemblies Sliceable

- No horizontal or vertical slice will separate
- Prevents ordering so can route in one pass

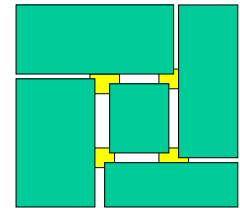


Penn ESE535 Spring 2009 -- DeHon

15

Switchbox Routing

- Box with 3 or 4 sides fixed

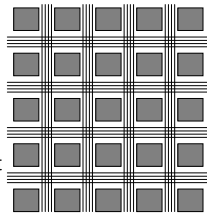


Penn ESE535 Spring 2009 -- DeHon

16

Gate Array → Channel

- Global route first
 - Decide which path each signal takes
 - Sequence of channels
 - Minimize congestion
 - Wires per channel segment

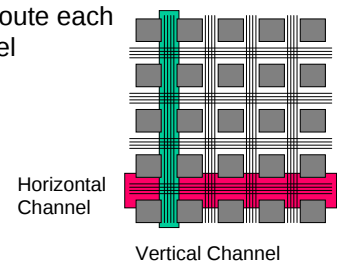


Penn ESE535 Spring 2009 -- DeHon

17

Gate Array → Channel

- Then Channel route each resulting channel

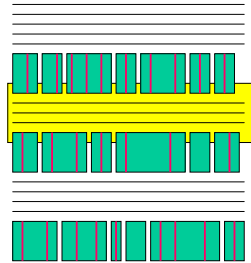


Penn ESE535 Spring 2009 -- DeHon

18

Std.Cell → Channel Route

- Plan feed through
- Channel route each row



Penn ESE535 Spring 2009 -- DeHon

19

Channel Routing

- Key subproblem in all variants
- Psuedo 1D problem
- **Given:** set of terminals on one or both sides of channel
- Assign to tracks to minimize channel width

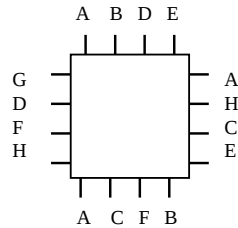


Penn ESE535 Spring 2009 -- DeHon

20

Switchbox Route

- Terminals on 4 sides
- Link up terminal



Penn ESE535 Spring 2009 -- DeHon

21

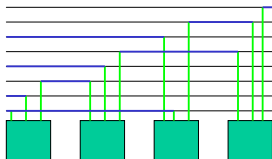
Channel Routing

Penn ESE535 Spring 2009 -- DeHon

22

Trivial Channel Routing

- Assign every net its own track
 - Channel width $> N$ (single output functions)
 - Chip bisection $\propto N \rightarrow$ chip area N^2

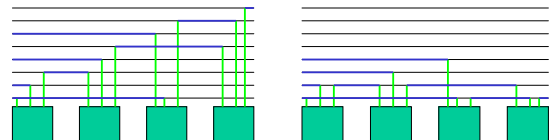


Penn ESE535 Spring 2009 -- DeHon

23

Sharing Tracks

- Want to Minimize tracks used
- Trick is to share tracks

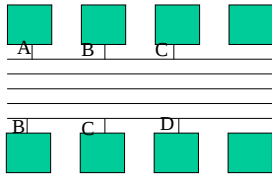


Penn ESE535 Spring 2009 -- DeHon

24

Not that Easy

- With Two sides
 - Even assigning one track/signal may not be enough

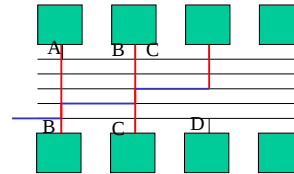


Penn ESE535 Spring 2009 -- DeHon

25

Not that Easy

- With Two sides
 - Even assigning one track/signal may not be enough



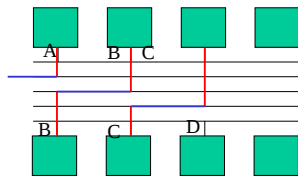
Bad assignment
Overlap:
A,B
B,C

Penn ESE535 Spring 2009 -- DeHon

26

Not that Easy

- With Two sides
 - Even assigning one track/signal may not be enough



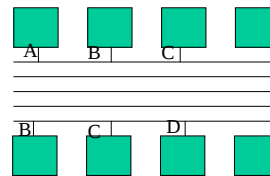
Valid assignment
avoids
overlap

Penn ESE535 Spring 2009 -- DeHon

27

Not that Easy

- With Two sides
 - Even assigning one track/signal may not be enough



*i.e. there are
vertical
constraints
on ordering*

Penn ESE535 Spring 2009 -- DeHon

28

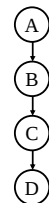
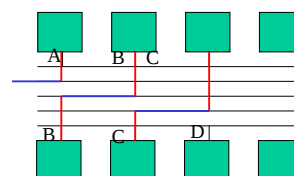
Vertical Constraints

- For vertically aligned pins:
 - With single “vertical” routing layer
 - Cannot have distinct top pins on a lower track than bottom pins
 - Leads to vertical overlap
 - Produces constraint that top wire be higher track than lower
 - Combine across all top/bottom pairs
 - Leads to a Vertical Constraint Graph (VCG)

Penn ESE535 Spring 2009 -- DeHon

29

VCG Example



Penn ESE535 Spring 2009 -- DeHon

30

Channel Routing Complexity

- With Vertical Constraints
 - Problem becomes NP-complete
- Without Vertical Constraints
 - Can be solved optimally
 - Tracks = maximum channel density
 - Greedy algorithm

Penn ESE535 Spring 2009 -- DeHon

31

No Vertical Constraints

Good for:

- Single-sided channel
 - (no top and bottom pins)
- Three layers for routing
 - Two vertical channels allow top and bottom pins to cross
 - May not be best way to use 3 layers...

Penn ESE535 Spring 2009 -- DeHon

32

Left-Edge Algorithm

1. Sort nets on leftmost end position
2. Start next lowest track; end=0
3. While there are unrouted nets with lowest left position > end of this track
 - Select unrouted net with lowest left position > end
 - Place selected net on this track
 - Update end position on this track to be end position of selected net
4. If nets remain, return to step 2

Greedy, optimal.

Penn ESE535 Spring 2009 -- DeHon

33

Example: Left-Edge

Note: nets (shown as letters here) show up as numbers in conv. channel routing file formats.

- Top: a b g b c d f
- Bottom: g d f e a c e
- Nets:
 - a:1—5
 - b:2—4
 - c:5—6
 - d:2—6
 - e:4—7
 - f:3—7
 - g:1—3

Penn ESE535 Spring 2009 -- DeHon

34

Example: Left-Edge

- Top: a b g b c d f
- Bottom: g d f e a c e
- Sort Left Edge:

– a:1—5	– a:1—5
– b:2—4	– g:1—3
– c:5—6	– b:2—4
– d:2—6	– d:2—6
– e:4—7	– f:3—7
– f:3—7	– e:4—7
– g:1—3	– c:5—6

Penn ESE535 Spring 2009 -- DeHon

35

Example: Left-Edge

- Top: a b g b c d f
- Bottom: g d f e a c e
- Sort Left Edge:

– a:1—5	– Track 0:
– g:1—3	– End 0
– b:2—4	– Add a:1—5
– d:2—6	– End 5
– f:3—7	
– e:4—7	
– c:5—6	

Penn ESE535 Spring 2009 -- DeHon

36

Example: Left-Edge

- Top: a b g b c d f
- Bottom: g d f e a c e
- Sort Left Edge:
 - Track 0: a:1—5
 - Track 1:
 - g:1—3
 - b:2—4
 - d:2—6
 - f:3—7
 - e:4—7
 - c:5—6
 - End 0
 - g:1—3
 - End 3
 - e: 4—7
 - End 7

Penn ESE535 Spring 2009 -- DeHon

37

Example: Left-Edge

- Top: a b g b c d f
- Bottom: g d f e a c e
- Sort Left Edge:
 - Track 0: a:1—5
 - Track 1: g:1—3, e:4—7
 - Track 2:
 - b:2—4
 - d:2—6
 - f:3—7
 - c:5—6
 - End 0
 - b:2—4
 - End 4
 - c:5—6
 - End 6

Penn ESE535 Spring 2009 -- DeHon

38

Example: Left-Edge

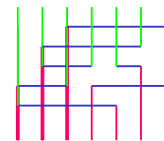
- Top: a b g b c d f
- Bottom: g d f e a c e
- Sort Left Edge:
 - Track 0: a:1—5
 - Track 1: g:1—3, 4:e—7
 - Track 2: b:2—4, c:5—6
 - Track 3: d:2—6
 - Track 4: f:3—7

Penn ESE535 Spring 2009 -- DeHon

39

Example: Left-Edge

- Top: a b g b c d f
- Bottom: g d f e a c e
- Track 0: a:1—5
- Track 1: g:1—3, e:4—7
- Track 2: b:2—4, c:5—6
- Track 3: d:2—6
- Track 4: f:3—7



Penn ESE535 Spring 2009 -- DeHon

40

Constrained Left-Edge

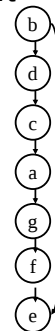
1. Construct VCG
2. Sort nets on leftmost end position
3. Start new track; end=0
4. While there are nets that have
 - ✓ No descendents in VCG
 - ✓ And left edge > end
 1. Place net on track and update end
 2. Delete net from list, VCG
5. If there are still nets left to route, return to 2

Penn ESE535 Spring 2009 -- DeHon

41

Example: Constrained Left-Edge

- Top: a b g b c d f
- Bottom: g d f e a c e
- Nets:
 - a:1—5
 - b:2—4
 - c:5—6
 - d:2—6
 - e:4—7
 - f:3—7
 - g:1—3
- Vertical Constraints
 - a→g
 - b→d
 - g→f
 - b→e
 - c→a
 - d→c
 - f→e



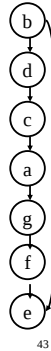
Penn ESE535 Spring 2009 -- DeHon

42

Example: ...

- Top: a b g b c d f
- Bottom: g d f e a c e
- Sort Left Edge:
 - a:1—5
 - g:1—3
 - b:2—4
 - d:2—6
 - f:3—7
 - e:4—7
 - c:5—6

- Track 0:
 - e:4—7
- Track 1:
 - f:3—7
- Track 2:
 - g:1—3
- Track 3:
 - a:1—5
- Track 4:
 - c:5—6
- Track 5:
 - d:2—6
- Track 6:
 - b:2—4

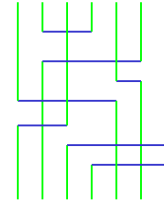


43

Penn ESE535 Spring 2009 -- DeHon

Example: Left-Edge

- Top: a b g b c d f
- Bottom: g d f e a c e
- Nets:
 - a:1—5
 - b:2—4
 - c:5—6
 - d:2—6
 - e:4—7
 - f:3—7
 - g:1—3

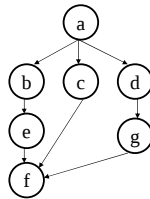


44

Penn ESE535 Spring 2009 -- DeHon

Example 2: ...

- Top: a a a b e d g c
- Bottom: b c d e f g f f
- Sort Left Edge:
 - b:1—4
 - a:1—3
 - c:2—8
 - d:3—6
 - e:4—5
 - f:5—8
 - c:6—7



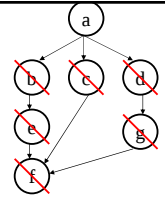
45

Penn ESE535 Spring 2009 -- DeHon

Example 2: ...

- Top: a a a b e d g c
- Bottom: b c d e f g f f
- Sort Left Edge:
 - ~~– b:1—4~~
 - ~~– a:1—3~~
 - ~~– c:2—8~~
 - ~~– d:3—6~~
 - ~~– e:4—5~~
 - ~~– f:5—8~~
 - ~~– g:6—7~~

- Track 0: f
- Track 1: c
- Track 2: e, g
- Track 3: b
- Track 4: d
- Track 5: a

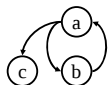


46

Penn ESE535 Spring 2009 -- DeHon

VCG Cycles

- Top: a a b
- Bottom: b c a
- VCG:

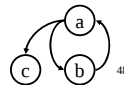


47

Penn ESE535 Spring 2009 -- DeHon

VCG Cycles

- No channel ordering satisfies VCG
- Must relax **artificial** constraint of single horizontal track per signal
- **Dogleg**: split horizontal run into multiple track segments
- In general, can reduce track requirements

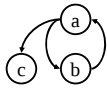


48

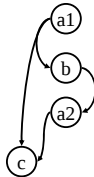
Penn ESE535 Spring 2009 -- DeHon

Dogleg Cycle Elimination

- Top: a a b
- Bottom: b c a
- VCG:



- Top: a1 a1/a2 b
- Bottom: b c a2
- VCG:

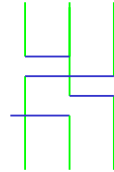
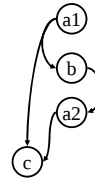


Penn ESE535 Spring 2009 -- DeHon

49

Dogleg Cycle Elimination

- Top: a1 a1/a2 b
- Bottom: b c a2
- VCG:



Penn ESE535 Spring 2009 -- DeHon

50

Dogleg Algorithm

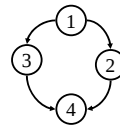
1. Break net into segments at pin positions
2. Build VCG based on segments
3. Run constrained on segments rather than full wires

Penn ESE535 Spring 2009 -- DeHon

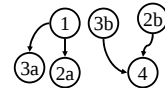
51

Dogleg Example (no cycle)

- Top: 1 1 2 - 2 3
- Bottom: 2 3 - 3 4 4



1 1 2a/2b - 2b 3b
2a 3a - 3a/b 4 4



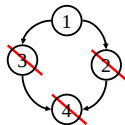
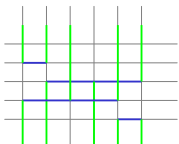
[note: switch to numbers for terminals]

Penn ESE535 Spring 2009 -- DeHon

52

No Dogleg

- Top: 1 1 2 - 2 3
- Bottom: 2 3 - 3 4 4

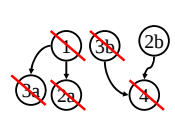
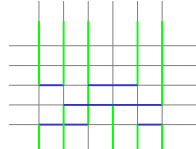


Penn ESE535 Spring 2009 -- DeHon

53

With Dogleg

- Top: 1 1 2a/2b - 2b 3b
- Bottom: 2a 3a - 3a/b 4 4

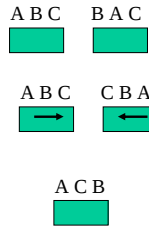


Penn ESE535 Spring 2009 -- DeHon

54

Other Freedoms

- Swap equivalent pins
 - E.g. nand inputs equivalent
- Mirror cells
 - if allowed electrically
- Choose among cell instances
 - Permute pins

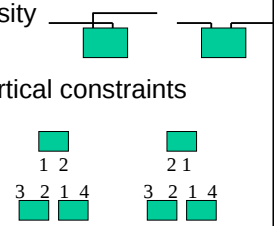


Penn ESE535 Spring 2009 -- DeHon

55

Exploit Freedom To

- Reduce channel density
- Reduce/Eliminate vertical constraints
 - Cycles
 - VCG height



Penn ESE535 Spring 2009 -- DeHon

56

Over The Cell

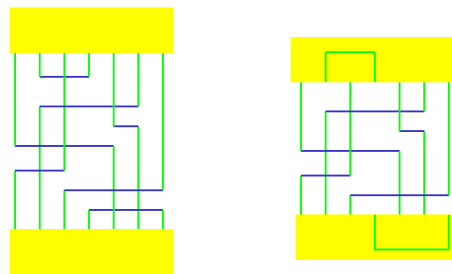
- Limit cell to lower metal
 - Maybe only up to M1
- Can route over with higher metal

Penn ESE535 Spring 2009 -- DeHon

57

Example: OTC

- Top: 0 1 6 1 2 3 5
- Bottom: 6 3 5 4 0 2 4

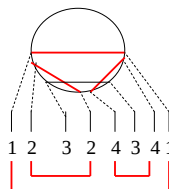


Penn ESE535 Spring 2009 -- DeHon

58

Over The Cell

- Compute maximal independent set
 - To find nets can be routed in 1 layer (planar) over cell
 - MIS can be computed in $O(n^2)$ time with dynamic programming
- Then route residual connections in channel
- Works on 2-metal if only M1 in cell
 - Feedthrus in M1



Penn ESE535 Spring 2009 -- DeHon

59

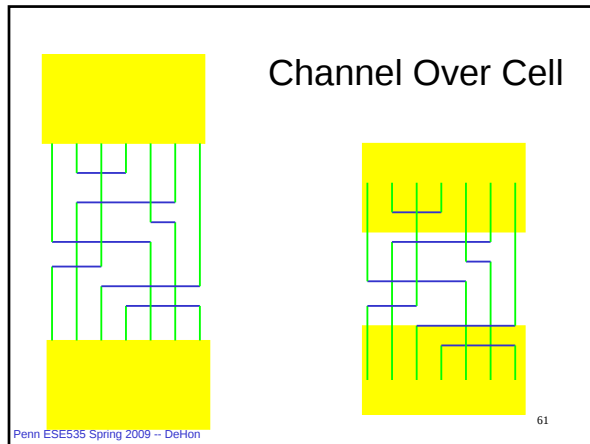
Multilayer

- With 3 layer
 - Can run channel over cells
 - Put Terminals in center of cell



Penn ESE535 Spring 2009 -- DeHon

60



- ### Route Over Cells
- If channel width < cell height
 - Routing completely on top of cells
 - If channel width > cell height
 - Cell area completely hidden under routing channel
 - More typical case
 - Especially for large rows
- Penn ESE535 Spring 2009 -- DeHon
- 62

- ### Summary
- Decompose Routing
 - Channel Routing
 - Left-Edge
 - Vertical Constraints
 - Exploiting Freedom
 - Dogleg, pin swapping
 - Routing over logic
- Penn ESE535 Spring 2009 -- DeHon
- 63

- ### Admin
- Reading for Monday online
- Penn ESE535 Spring 2009 -- DeHon
- 64

- ### Big Ideas
- Decompose Problem
 - Divide and conquer
 - Interrelation of components
 - Structure: special case can solve optimally
 - Technique: Greedy algorithm
 - Use greedy as starting point for more general algorithm
- Penn ESE535 Spring 2009 -- DeHon
- 65