ESE534:
Computer Organization

Day 16: March 19, 2012
Interconnect 3: Richness

Penn

Last Time

- Rent's Rule
  - And its implications
- Superlinear growth rate of interconnect
  p>0.5
  - $\rightarrow$ Area growth $\Omega(N^{2p})$

2

Today

- How rich should interconnect be?
  - specifics of understanding interconnect
  - methodology for attacking these kinds of questions

3

Now What?

- There is structure (locality)
- Rent characterizes locality

- How rich should interconnect be?
  - Allow full utilization of compute units?
  - What is most area efficient?
 - $\rightarrow$Need to model requirements and area impact
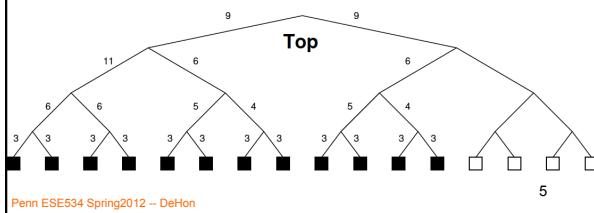
4

Preclass 1 and 2

- Wire count?

5

Preclass 1 and 2

- Wire count?

6

1

Preclass 1 and 2

- Wire count?

Bottom

7
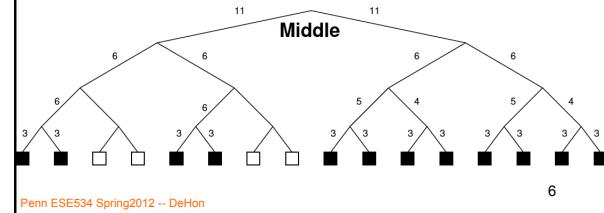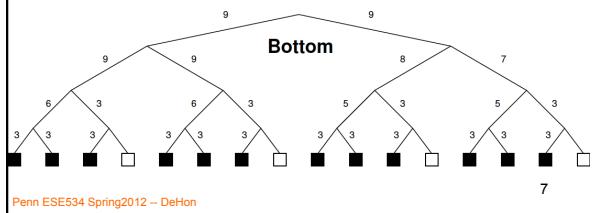
Step 1: Build Architecture Model

- Assume geometric growth
- Pick parameters: Build architecture can tune
  - C
  - p

8
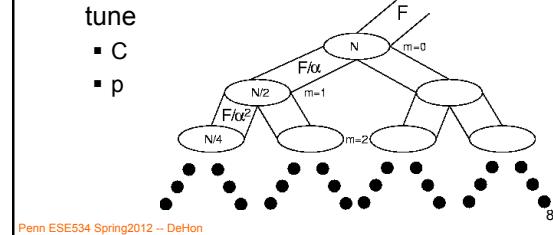
Tree of Meshes
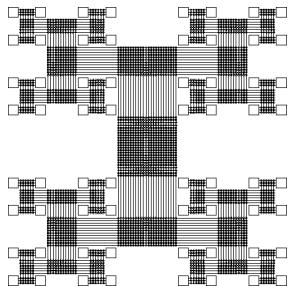
- Natural model is hierarchical
- Restricted internal bandwidth
- Can match to model

9

Parameterize C
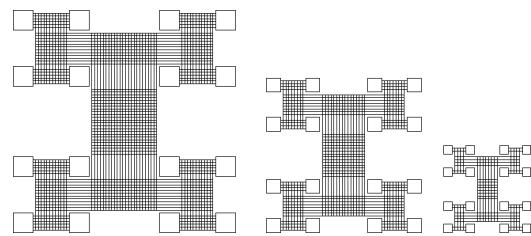
10

Parameterize Growth by p

- What are IO schedules? (preclass 3)
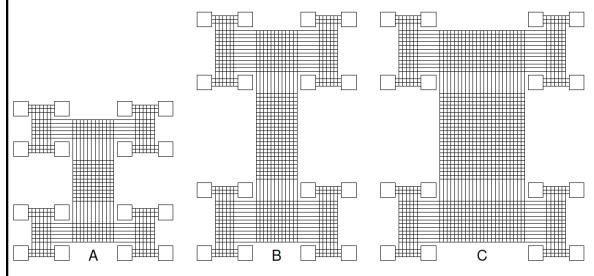
11

Preclass 5

- What are IO schedules according to Rent for particular p's?
  - p=1/2
  - p=2/3
  - p=3/4

$$IO = c \, N^p$$

12

2

Parameterize p

- What is p for each network? (preclass 6)
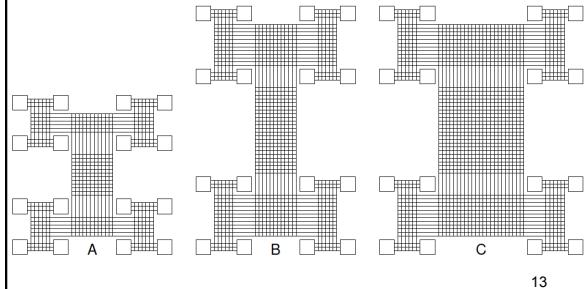
A   B   C

Parameterize Growth

2:1   1:1

1:1

1:1   2:1   2:1   1:1   2:1   2:1

2:1   p=0.5   2:1   p=0.67   2:1   p=0.75

(2 1)*   (2 2 1)*   (2 2 2 1)*
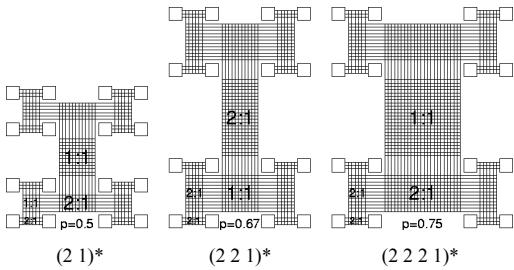
Step 2: Area Model

- Need to know effect of architecture parameters on area (costs)
  - focus on dominant components
    - wires
    - switches
    - logic blocks(?)

Area Parameters

- A_{logic} = 10K F^2
- A_{sw} = 625 F^2
- Wire Pitch = 4 F

Switchbox Population

- Full population is excessive (next lecture)
- **Hypothesis:** linear population adequate
  - still to be (dis)proven
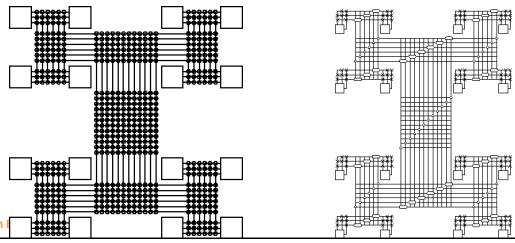
"Cartoon" VLSI Area Model
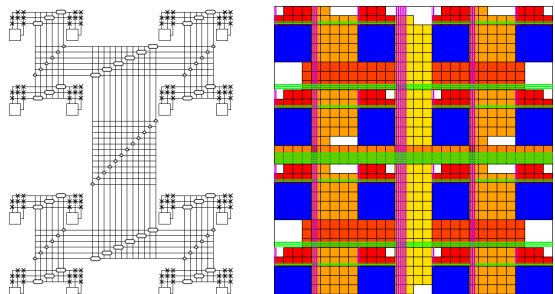
(Example artificially small for clarity)

Larger "Cartoon"

1024 LUT
Network

P=0.67

LUT
Area 3%

Effects of P on Area

P=0.5   P=0.67   P=0.75

1024 LUT Area Comparison

Effects of P on Capacity

- - - p=0.50
——— p=0.67
······ p=0.75

N_{LUT} [x1024]

Area [in $M\lambda^2$]

## Step 3: Characterize Application Requirements

- Identify representative applications.
  - Today: IWLS93 logic benchmarks
- How much structure there?
- How much variation among applications?

Application Requirements

Compare
Problem 1

Spectral Recursive Bisection

Max: C=7, P=0.68   Avg: C=5, P=0.72

Benchmark Wide

Benchmark Wide I/O vs. subgraph Size

Benchmark Parameters

25

Complication

- Interconnect requirements
  - vary among applications
- Interconnect richness
  - has **large** effect on area
- What is effect of architecture/application mismatch?
  – Interconnect too rich?
  – Interconnect too poor?

26

Interconnect Mismatch in Theory
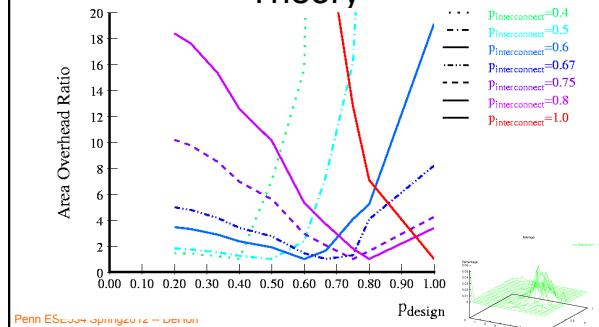
Step 4: Assess Resource Impact

- Map designs to parameterized architecture
- Identify architectural resource required

**Compare**: mapping to k-LUTs; LUT count vs. k.

28

Mapping to Fixed Wire Schedule

- Easy if need fewer wires than Net

- If need more wires than net, must depopulate to meet interconnect limitations.
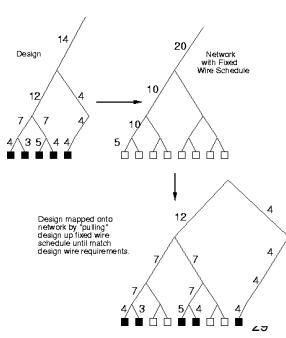

Preclass 4

- Smallest Network that Top graph fits on?

30

5
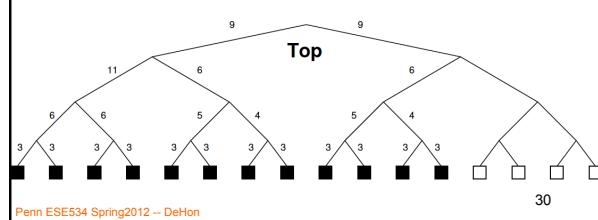
Mapping to Fixed-WS

- Better results if "reassociate" rather than keeping original subtrees.
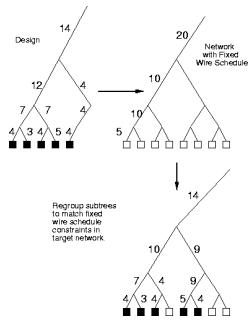
Preclass 4

- Smallest Network that Middle graph fits on?
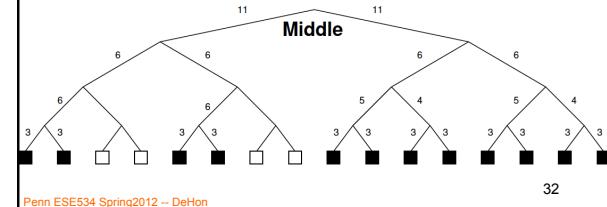
32

## Middle vs. Top

33
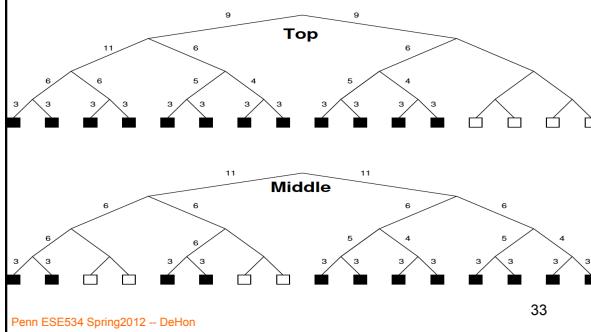
Observation

- Don't really want a "bisection" of LUTs
  - subtree filled to capacity by *either* of
    - LUTs
    - root bandwidth
  - May be profitable to cut at some place other than midpoint
    - not require "balance" condition
 - "Bisection" should account for both LUT and wiring limitations

34

Preclass 4

- Smallest Network that Bottom graph fits on?
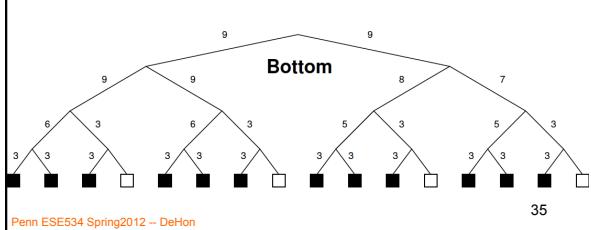
35

Middle vs. Bottom

36
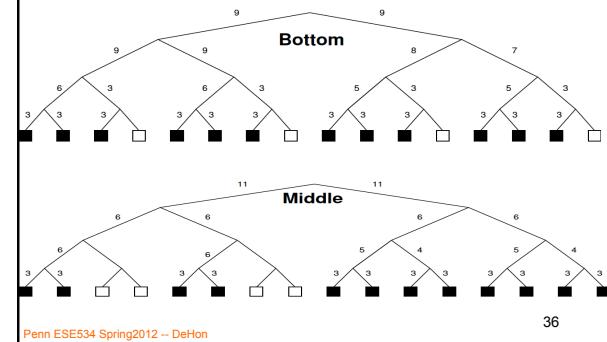
Challenge

- Not know where to cut design
  - not knowing when wires will limit subtree capacity

Brute Force Solution

- Explore all cuts
  - start with all LUTs in group
  - consider "all" balances
  - try cut
  - Recurse

- Viable?

Brute Force

- Too expensive
- Exponential work

- …viable if solving same subproblems

Simplification

- Single linear ordering
- Partitions = pick split point on ordering
- Reduce to finding cost of [start,end] ranges (subtrees) within linear ordering
- Only n^2 such subproblems
- Can solve with dynamic programming

Dynamic Programming

- Just one possible "heuristic" solution to this problem
  - not optimal
  - dependent on ordering
  - sacrifices ability to reorder on splits to avoid exponential problem size

- Opportunity to find a better solution here...

Ordering LUTs

- Another problem
  - lay out gates in 1D line
  - minimize sum of squared wire length
    - tend to cluster connected gates together
  - Is solvable mathematically for optimal
    - Eigenvector of connectivity matrix

- Use this 1D ordering for our linear ordering

Mapping Results

.3

## Step 5: Apply Area Model

- Assess impact of resource results

44

Resources × Area Model ⇒ Area

45

Net Area

46

Picking Network Design Point

Minimize Objective	params C	P	Sigma rel area	LUT Util.
relative area	6	0.6	1.23	0.87
area with full util	10	0.75	2.98	1.00

Don't optimize for 100% compute util. (100% yield)
…also don't optimize for highest peak.

47

What about a single design?

48

8
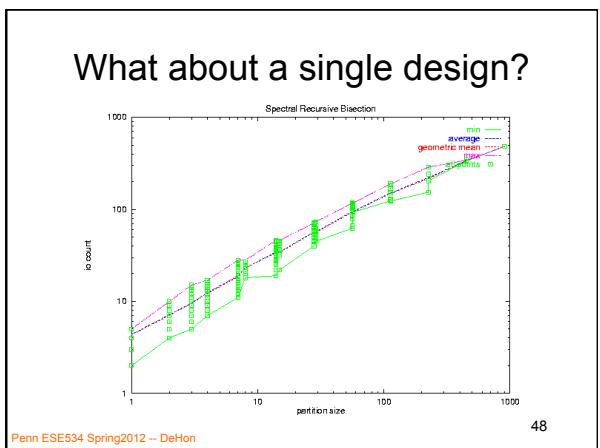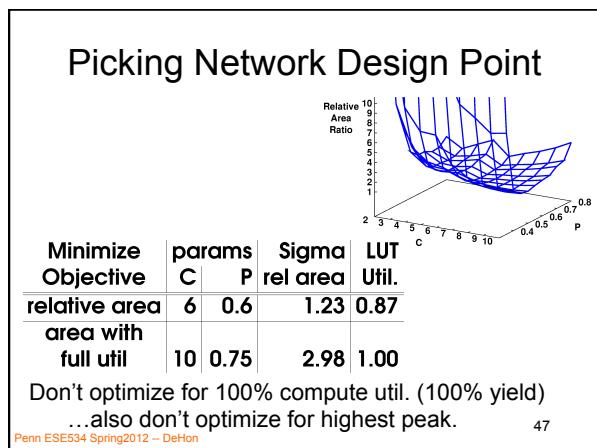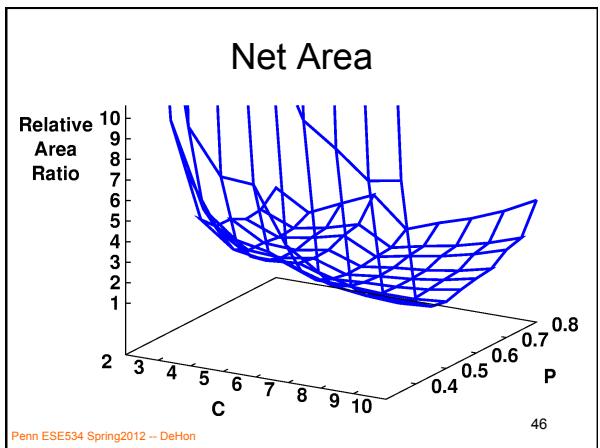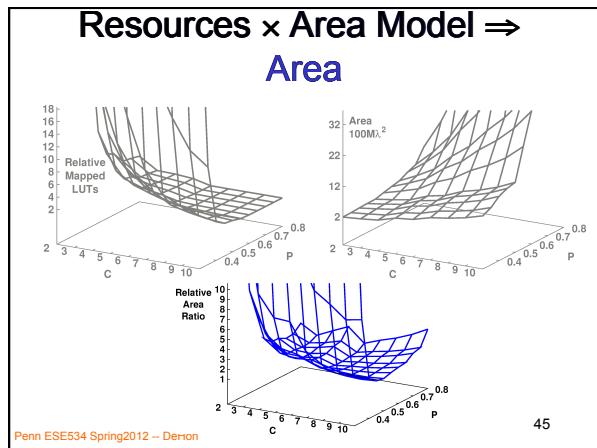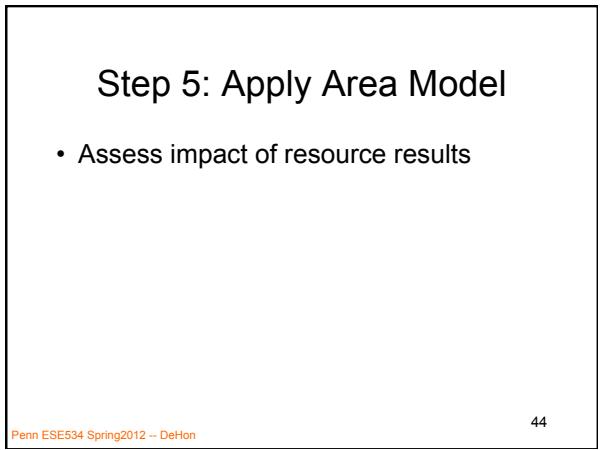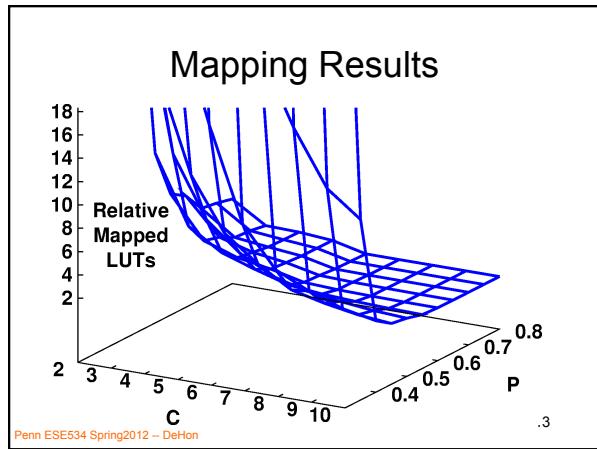
LUT Utilization predict Area?

C=6.0 Lsize Area Comparison

Single design

LUT Usage —
Relative Area ----

percentage

1
0.8
0.6
0.4
0.2
0

0.3   0.4   0.5   P 0.6   0.7   0.8
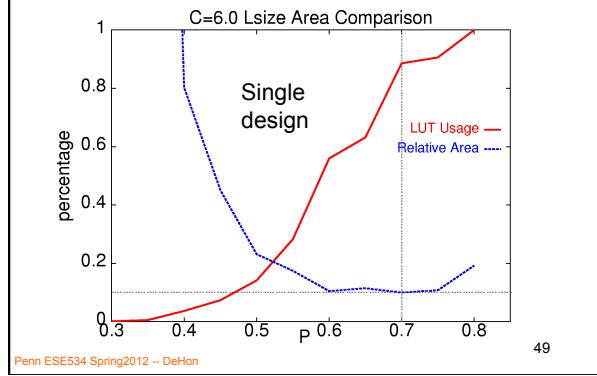
Methodology

1. Architecture model (parameterized)
2. Cost model
3. Important task characteristics
4. Mapping Algorithm
   – Map to determine resources
5. Apply cost model
6. Digest results
   – find optimum (multiple?)
   – understand conflicts (avoidable?)

Admin

- Reading for Wednesday online
- HW8 out

Big Ideas
[MSB Ideas]

- Interconnect area dominates logic area
- Interconnect requirements vary
  – among designs
  – within a single design
- To minimize area
  – focus on using dominant resource (interconnect)
  – may underuse non-dominant resources (LUTs)

Big Ideas
[MSB Ideas]

- Two different resources here
  – compute, interconnect
- Balance of resources required varies among designs (even within designs)
- Cannot expect full utilization of every resource
- Most area-efficient designs may *waste* some compute resources (cheaper resource)

9