ESE534: Computer Organization

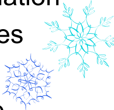
Day 16: March 24, 2010
Component-Specific Mapping

Previously

- Increasing variation with scaling (Day 8)
- Row sparing in memories (Day 12, HW6)
- Disk-drive model for mapping out defects in software (Day 12)
- PLAs for logic (last time)
- LUTs + Interconnect (Day 14)

Today

- Defect and Variation Tolerance
  for Compute and Interconnect
- High defect rates and extreme variation
- Nanoscale chips are like snowflakes
  ➔ Each one is unique.
- Application mapping tailored to the component



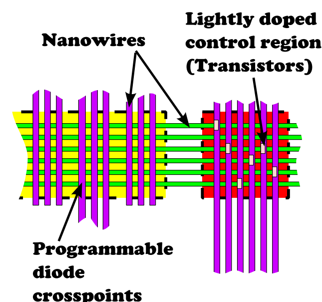
Agenda

- nanoPLA
  - Introduction
  - Tolerating crosspoint defects
  - Tolerating extreme V_{th} variation
- FPGA
  - Tolerating defects
  - Lightweight, load-time defect avoidance

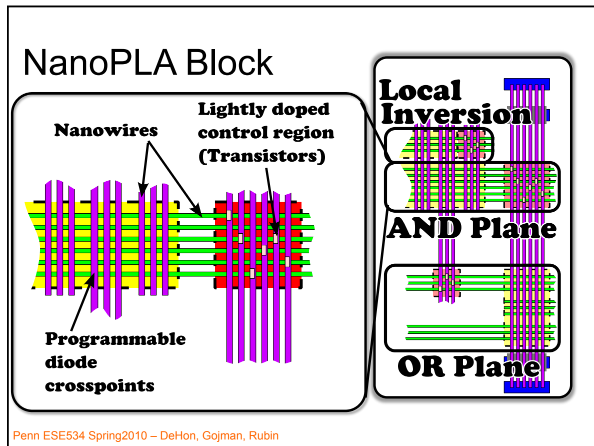
nanoPLA

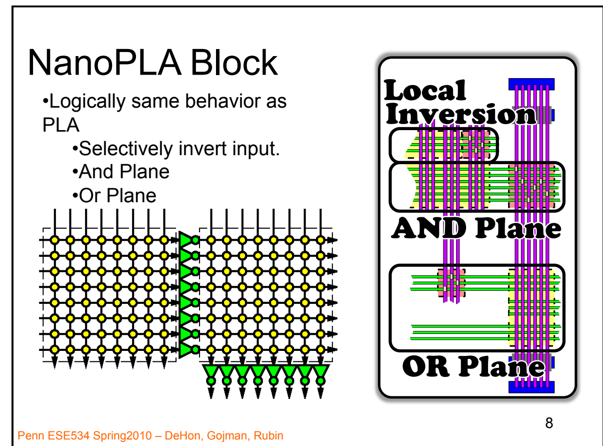
Architecture

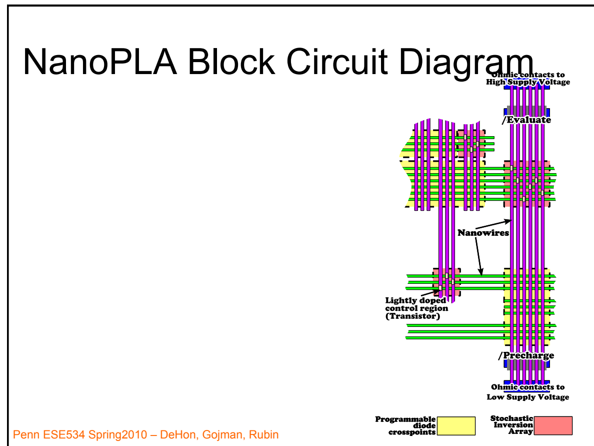
NanoPLA Plane

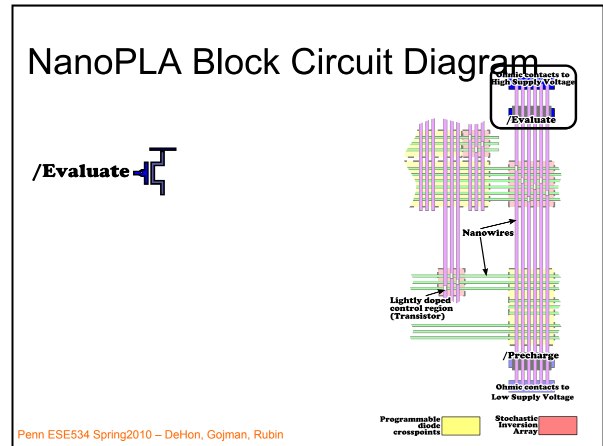


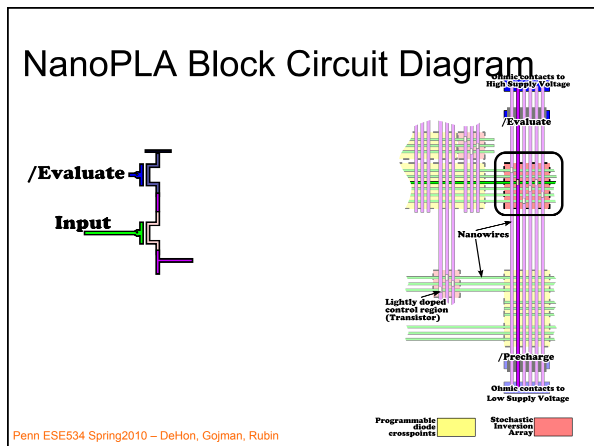
- Smallest unit of computation
- Built Bottom-Up from nanowires
- Diode-Programmable region
  - Memory cell area == Wire crossing area
  - Molecular
  - Amorphous Si
- Stochastic FET restore region
  - Core-shell nanowires with doped region

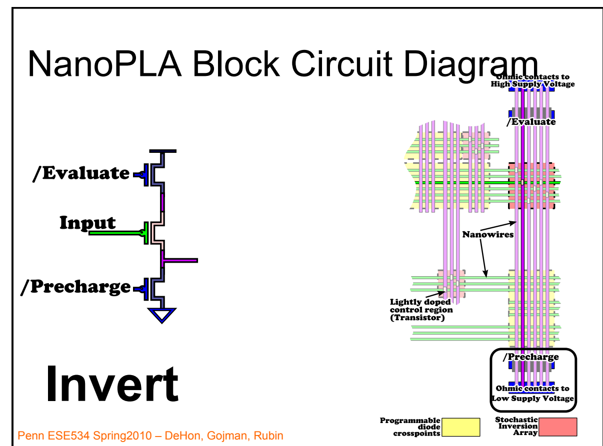

NanoPLA Block
Lightly doped control region (Transistors)
Nanowires
Programmable diode crosspoints
Local Inversion
AND Plane
OR Plane
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

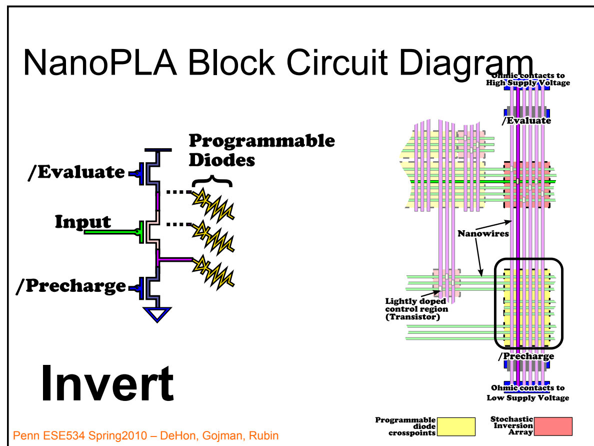

NanoPLA Block
•Logically same behavior as PLA
•Selectively invert input.
•And Plane
•Or Plane
Local Inversion
AND Plane
OR Plane
8
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin


NanoPLA Block Circuit Diagram
Ohmic contacts to High Supply Voltage
/Evaluate
Nanowires
Lightly doped control region (Transistor)
/Precharge
Ohmic contacts to Low Supply Voltage
Programmable diode crosspoints
Stochastic Inversion Array
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

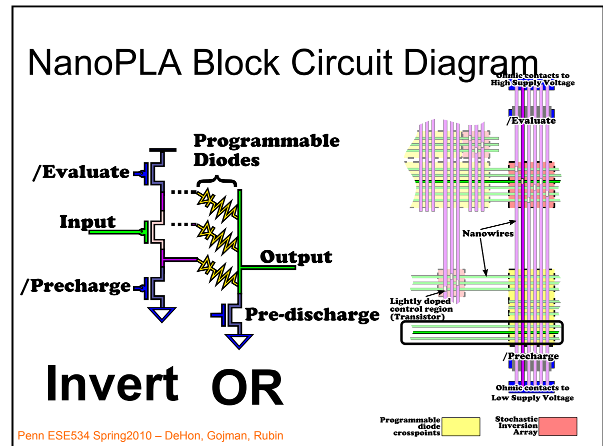

NanoPLA Block Circuit Diagram
/Evaluate
Ohmic contacts to High Supply Voltage
Nanowires
Lightly doped control region (Transistor)
/Precharge
Ohmic contacts to Low Supply Voltage
Programmable diode crosspoints
Stochastic Inversion Array
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin


NanoPLA Block Circuit Diagram
/Evaluate
Input
Ohmic contacts to High Supply Voltage
Nanowires
Lightly doped control region (Transistor)
/Precharge
Ohmic contacts to Low Supply Voltage
Programmable diode crosspoints
Stochastic Inversion Array
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

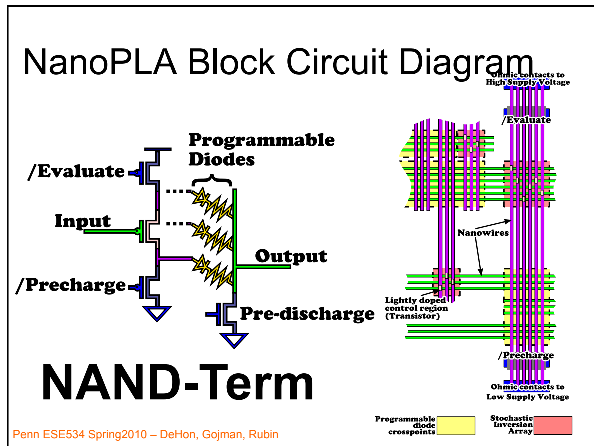

NanoPLA Block Circuit Diagram
/Evaluate
Input
/Precharge
Invert
Ohmic contacts to High Supply Voltage
Nanowires
Lightly doped control region (Transistor)
Ohmic contacts to Low Supply Voltage
Programmable diode crosspoints
Stochastic Inversion Array
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin



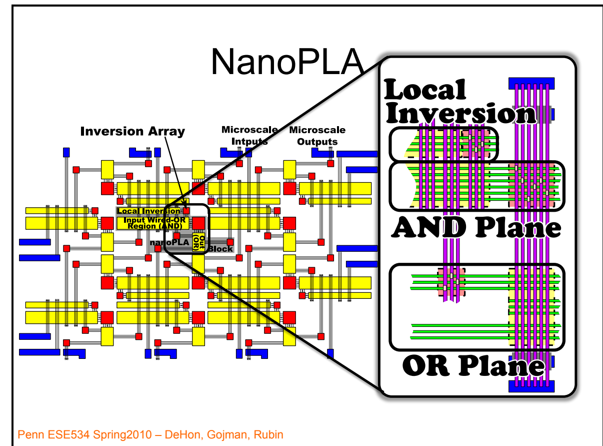
NanoPLA Block Circuit Diagram
Programmable Diodes
/Evaluate
Input
/Precharge
Invert
Ohmic contacts to High Supply Voltage
/Evaluate
Nanowires
Lightly doped control region (Transistor)
/Precharge
Ohmic contacts to Low Supply Voltage
Programmable diode crosspoints
Stochastic Inversion Array
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin



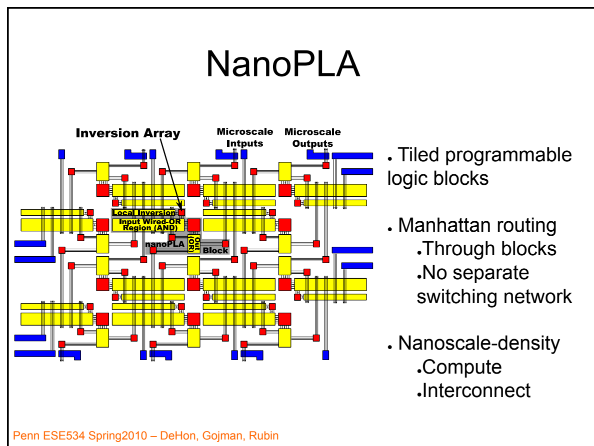
NanoPLA Block Circuit Diagram
Programmable Diodes
/Evaluate
Input
Output
/Precharge
Pre-discharge
Invert OR
Ohmic contacts to High Supply Voltage
/Evaluate
Nanowires
Lightly doped control region (Transistor)
/Precharge
Ohmic contacts to Low Supply Voltage
Programmable diode crosspoints
Stochastic Inversion Array
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin



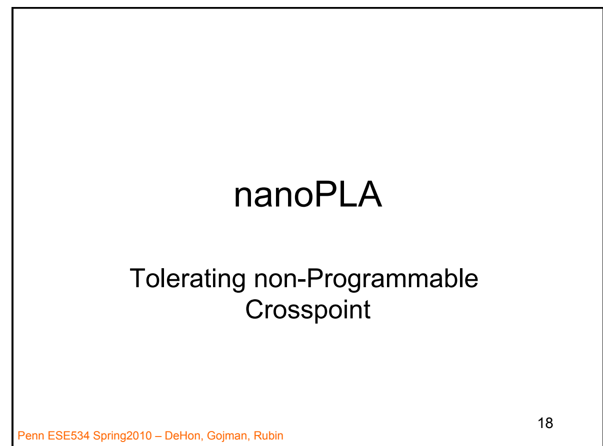
NanoPLA Block Circuit Diagram
Programmable Diodes
/Evaluate
Input
Output
/Precharge
Pre-discharge
NAND-Term
Ohmic contacts to High Supply Voltage
/Evaluate
Nanowires
Lightly doped control region (Transistor)
/Precharge
Ohmic contacts to Low Supply Voltage
Programmable diode crosspoints
Stochastic Inversion Array
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin



NanoPLA
Inversion Array
Microscale Intputs
Microscale Outputs
Local Inversion
AND Plane
OR Plane
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin



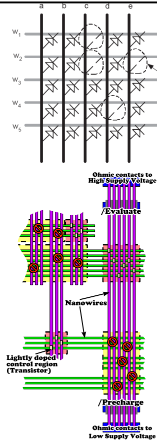
NanoPLA
Inversion Array
Microscale Intputs
Microscale Outputs
Local Inversion
Input Wired-OR Region (AND)
nanoPLA
Block
• Tiled programmable logic blocks
• Manhattan routing
•Through blocks
•No separate switching network
• Nanoscale-density
•Compute
•Interconnect
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin



nanoPLA
Tolerating non-Programmable Crosspoint
18
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Crosspoint Defects

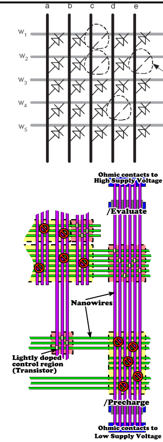
- Crosspoint junctions may be nonprogrammable
  - *E.g.* the first 8x8 from Hewlett-Packard had 85% programmable crosspoints
- Preclass:
  - 100 junctions
  - 5% stuck-open rate
  - P_{xpoint}=0.95
  - P_{row} ?



Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Crosspoint Defects

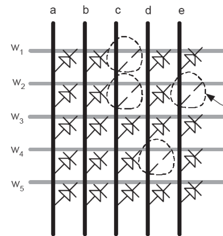
- Using P_{row} just calculated.
- How many perfect wires can we expect (on average) in an array of 100 wires?
- How will row sparing work here?
- What kind of overhead would row sparing require?



Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Idea 1: Don't need all junctions

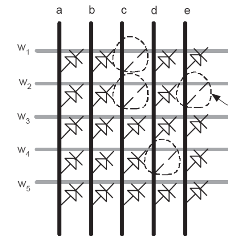
- F=a+b+e
- Can use wire w_4
- Even though has missing crosspoints
- Preclass 2:
  - F require 3 good crosspoints
  - P_{good}?



Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Idea 2: Have many wires to choose from

- F=a+b+e
- Cannot use w_2
- Can use any of the rest
- Probability of at least one match is very high
- Preclass 3
  - Use P_{good} from Preclass 2
  - P(1 of 5)=?



Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Mapping Entire Array

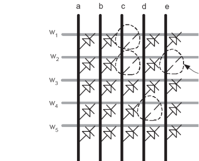
- Need to map all 100 functions
- Assuming $P_{1\text{-of-}5}$ from previous, what's the probability can find a match for all 100?
- In practice have 100 to choose from not 5
- What can we do if cannot find a match?

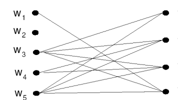
23

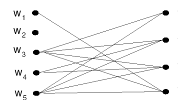
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

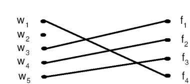
Crosspoint Defects

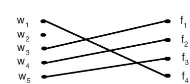
- Tolerate by **matching** nanowire junction programmability with pterm needs











Naeimi/DeHon, FPT2004

Design and Test of Computers, July-August 2005

Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Crosspoint Defect Rate Impact

- Toronto20 benchmark circuits
  - Typical 10k "gate" logic designs

Design	P_j 0.85	0.9	0.95	1
alu4	1.81	1.64	1.00	1.00
apex2	1.19	1.19	1.00	1.00
apex4	1.30	1.16	1.00	1.00
bigkey	1.00	1.00	1.00	1.00
clma	1.00	1.00	1.00	1.00
des	1.00	1.00	1.00	1.00
dsip	1.00	1.00	1.00	1.00
elliptic	1.00	1.00	1.00	1.00
ex1010	3.81	2.15	1.00	1.00
ex5p	1.00	1.00	1.00	1.00
frisc	1.00	1.00	1.00	1.00
misex3	1.31	1.31	1.00	1.00
pdc	4.75	1.79	1.00	1.00
s298	1.84	1.84	1.00	1.00
seq	1.20	1.12	1.00	1.00
spla	3.46	1.83	1.00	1.00

Matching Lesson

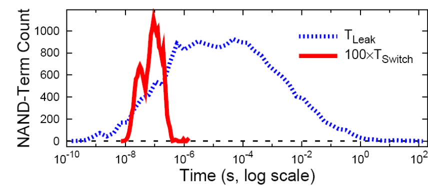
- Can use almost all wires
  - @ 5% defects, need **no** extra wires
- …despite the fact that almost all wires are imperfect
  - P_{row} from preclass 1
- Trick is to figure out where an imperfect wire is "good enough"

nanoPLA

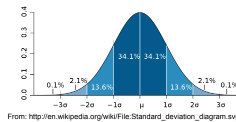
Tolerating Extreme V_{th} Variation

Challenge

- High variation of V_{th} means some devices leak faster than others evaluate!



Preclass 4



From: http://en.wikipedia.org/wiki/File:Standard_deviation_diagram.svg

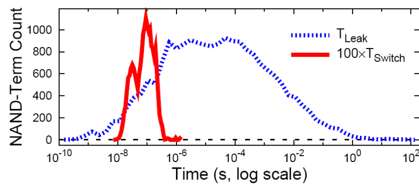
- Sampling nanowires from a distribution
- What range of nanowire V_{th}'s must we tolerate?
- 4c: In array of 100 nanowires, what is the probability that there is one nanowire nσ above mean and one nσ below?
  - σ = 1?
  - σ = 2?
  - σ = 3?

Preclass

- Problem 4d: How many σ out should we expect in a collection of 100 arrays?
- Problem 5:
  - $I(Von, V_{th}=V_{th}+n\sigma) = ?$
  - $I(Voff, V_{th}=V_{th}-n\sigma) = ?$
- What is σ/V_{th} given in example?

Challenge

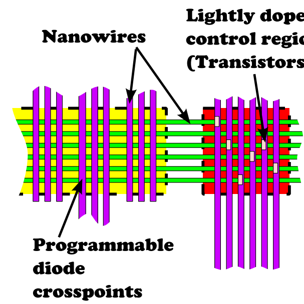
- High variation of V_{th} means some devices leak faster than others evaluate!



Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

31

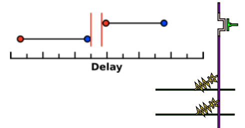
NanoPLA Physical Variation



- Predominantly Random Variation
- Due to Bottom-up manufacturing process
- Variation in:
  - Dopant fluctuations
  - NanoWire alignment
  - NanoWire dimensions
  - Number of state-holding elements / junction
- Variation in V_{th}, R and C
  - Independent Gaussian distributions

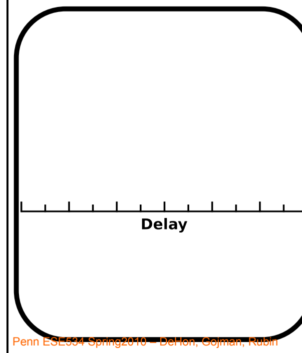
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

NanoPLA Defect Model



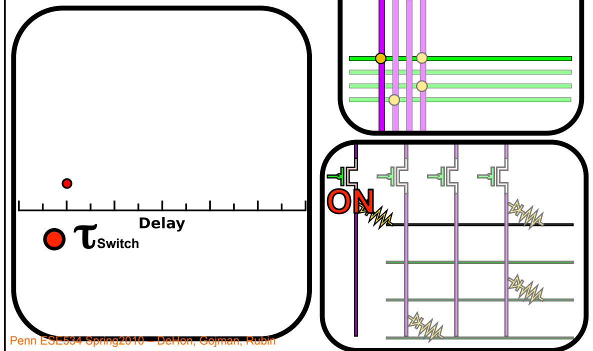
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Defect Model



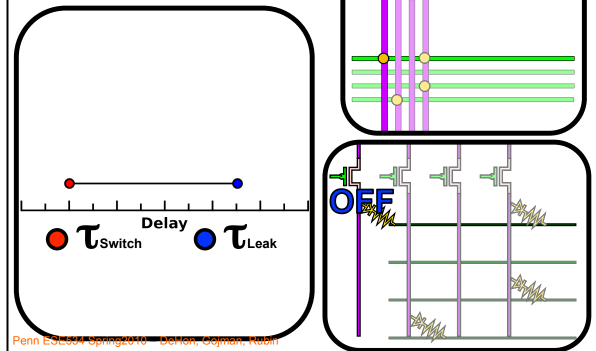
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Defect Model

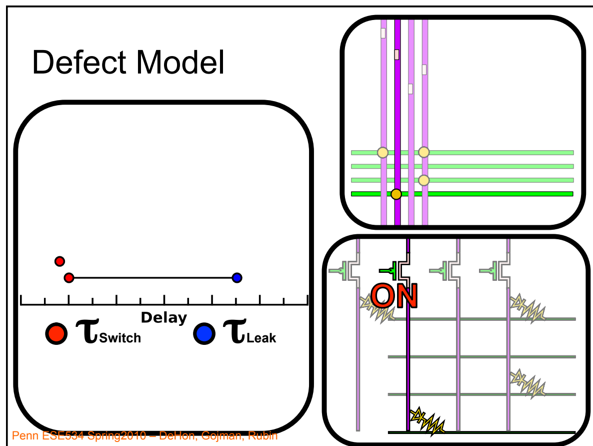


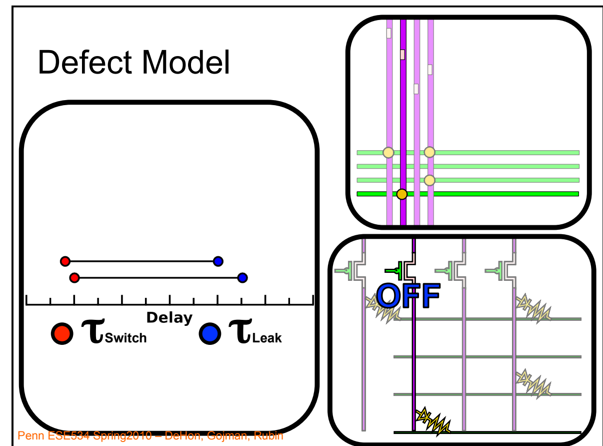
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

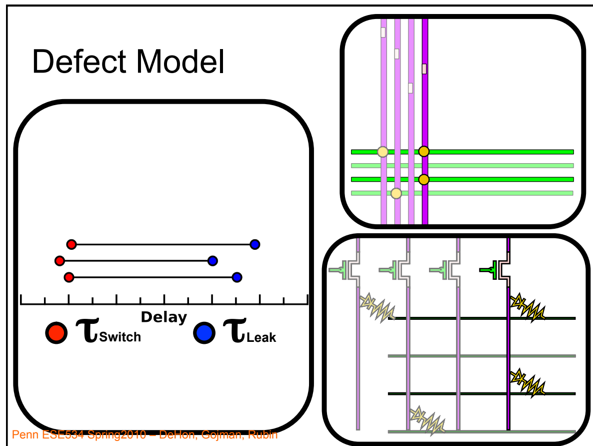
Defect Model

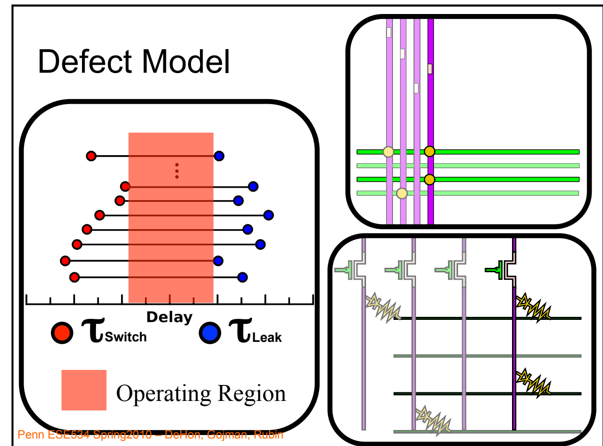


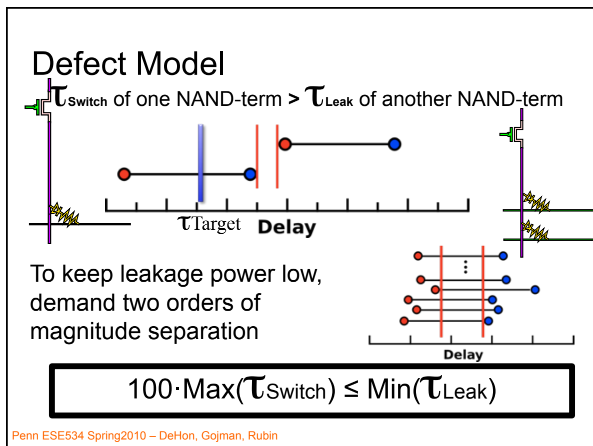
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

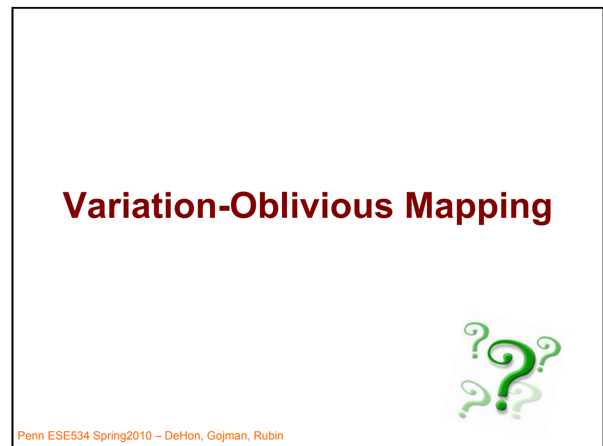

Defect Model
Delay
ON
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin


Defect Model
Delay
OFF
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin


Defect Model
Delay
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

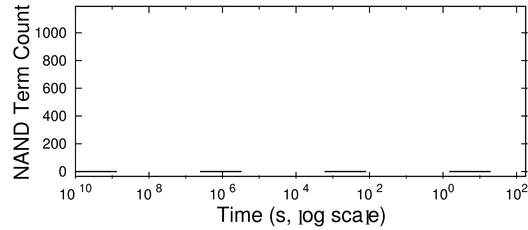

Defect Model
Delay
Operating Region
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin


Defect Model
τSwitch of one NAND-term > τLeak of another NAND-term
τTarget
Delay
To keep leakage power low, demand two orders of magnitude separation
100·Max(τSwitch) ≤ Min(τLeak)
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin


Variation-Oblivious Mapping
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Variation-Oblivious Mapping

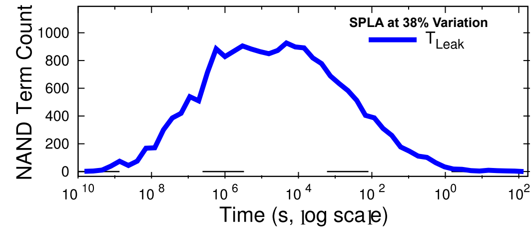
Perform Detailed Mapping Oblivious to the Characteristics of the underlying Resources.



Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Variation-Oblivious Mapping

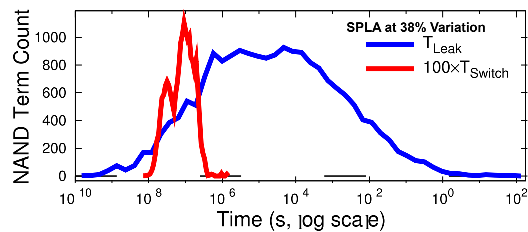
Perform Detailed Mapping Oblivious to the Characteristics of the underlying Resources.



Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Variation-Oblivious Mapping

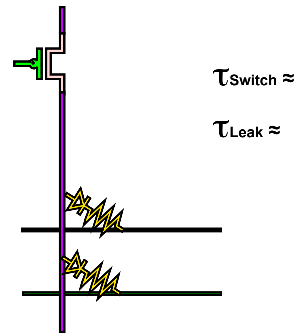
Perform Detailed Mapping Oblivious to the Characteristics of the underlying Resources.



Defective Mapping: 100·Max(τSwitch) ≤ Min(τLeak) not met

Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Timing Model

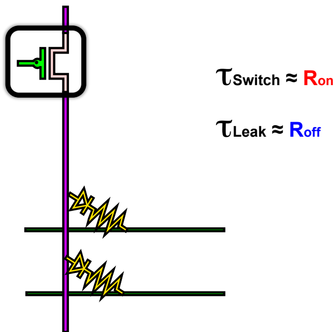


$\tau_{Switch} \approx$

$\tau_{Leak} \approx$

Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Timing Model

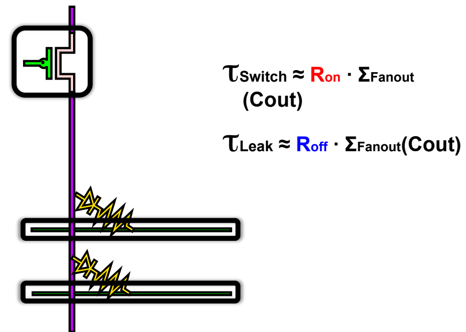


$\tau_{Switch} \approx R_{on}$

$\tau_{Leak} \approx R_{off}$

Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

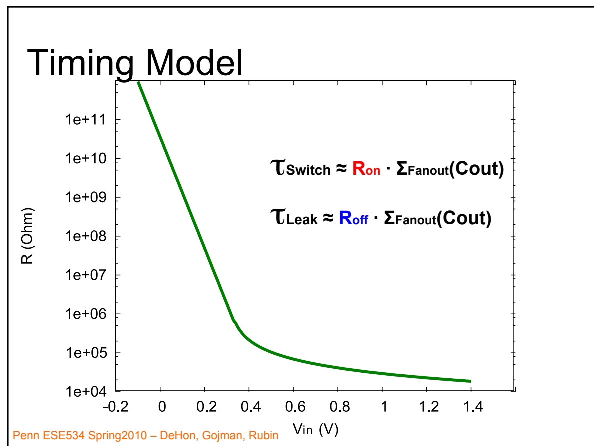
Timing Model

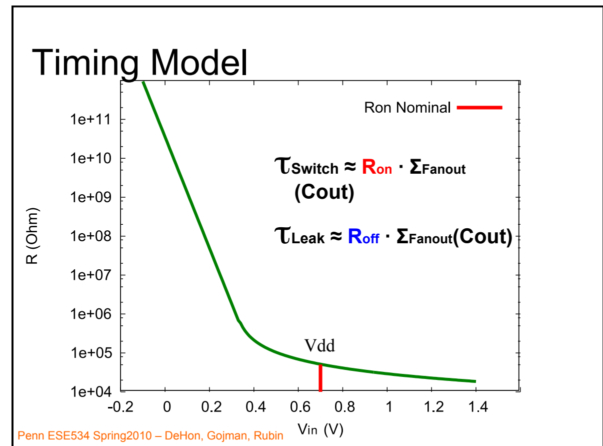


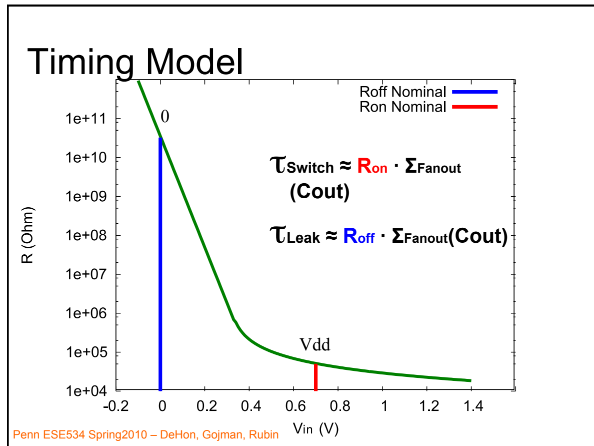
$\tau_{Switch} \approx R_{on} \cdot \Sigma_{Fanout}(Cout)$

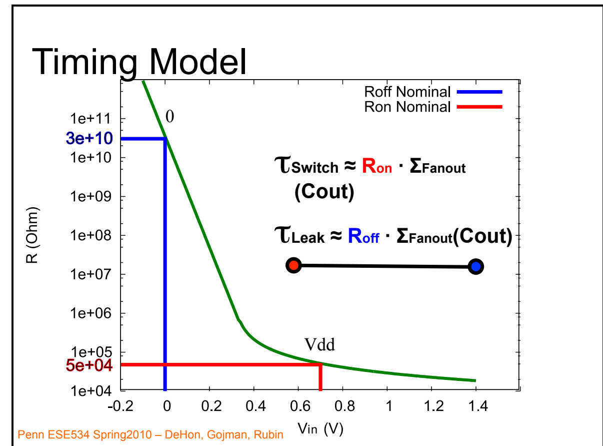
$\tau_{Leak} \approx R_{off} \cdot \Sigma_{Fanout}(Cout)$

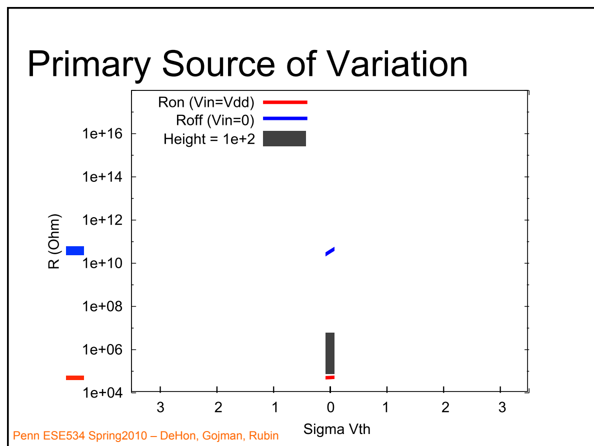
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

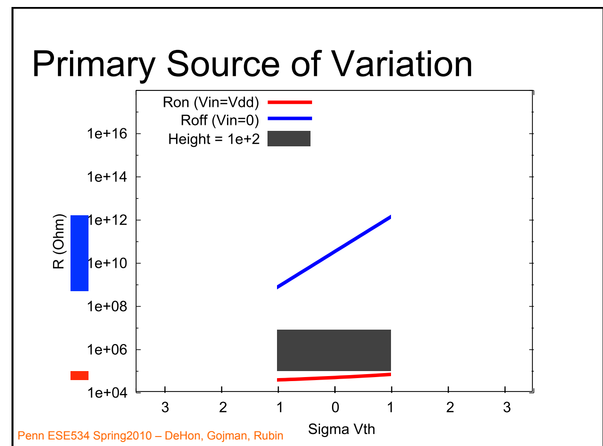

Timing Model
τSwitch ≈ Ron · ΣFanout(Cout)
τLeak ≈ Roff · ΣFanout(Cout)
R (Ohm)
Vin (V)
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

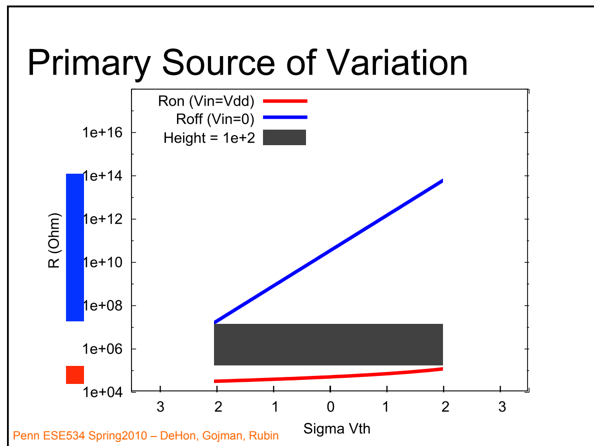

Timing Model
Ron Nominal
τSwitch ≈ Ron · ΣFanout (Cout)
τLeak ≈ Roff · ΣFanout(Cout)
Vdd
R (Ohm)
Vin (V)
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

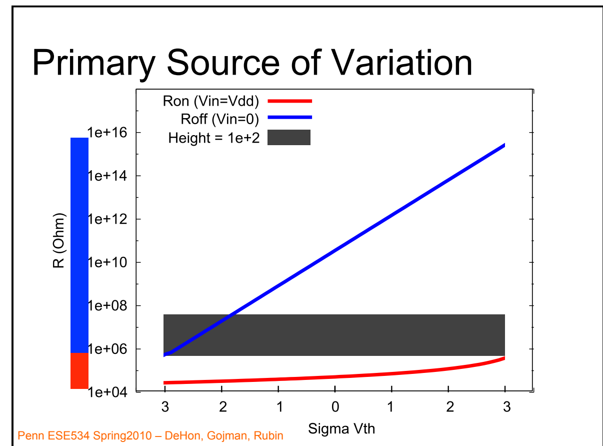

Timing Model
Roff Nominal
Ron Nominal
τSwitch ≈ Ron · ΣFanout (Cout)
τLeak ≈ Roff · ΣFanout(Cout)
0
Vdd
R (Ohm)
Vin (V)
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

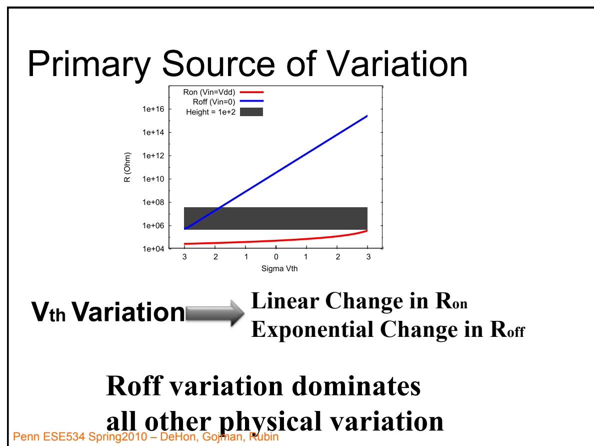

Timing Model
Roff Nominal
Ron Nominal
τSwitch ≈ Ron · ΣFanout (Cout)
τLeak ≈ Roff · ΣFanout(Cout)
3e+10
5e+04
0
Vdd
R (Ohm)
Vin (V)
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

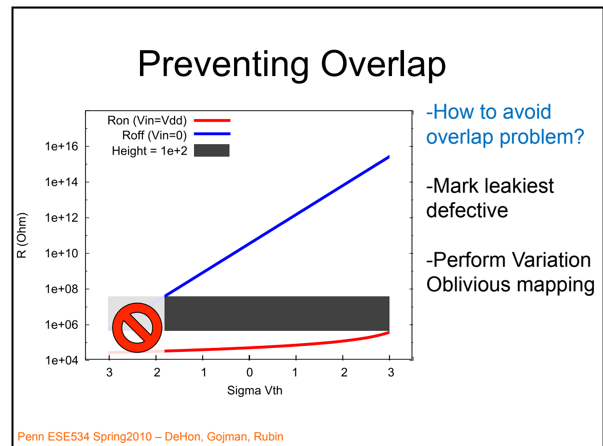

Primary Source of Variation
Ron (Vin=Vdd)
Roff (Vin=0)
Height = 1e+2
R (Ohm)
Sigma Vth
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

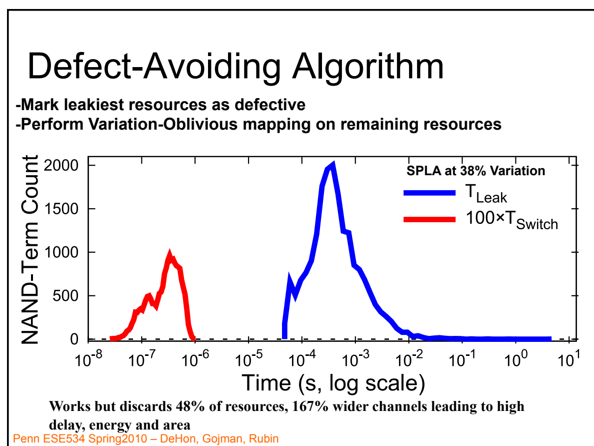

Primary Source of Variation
Ron (Vin=Vdd)
Roff (Vin=0)
Height = 1e+2
R (Ohm)
Sigma Vth
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

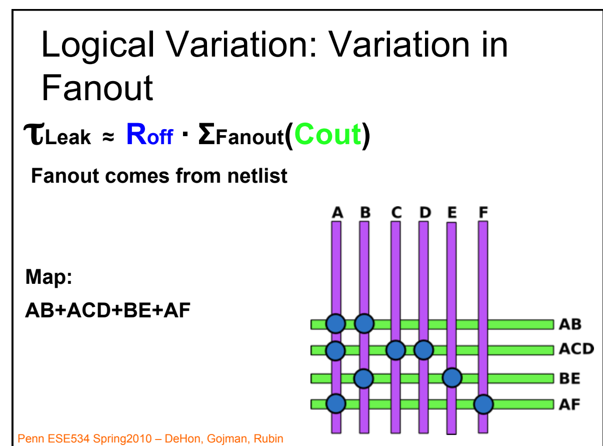

Primary Source of Variation
Ron (Vin=Vdd)
Roff (Vin=0)
Height = 1e+2
R (Ohm)
Sigma Vth
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin


Primary Source of Variation
Ron (Vin=Vdd)
Roff (Vin=0)
Height = 1e+2
R (Ohm)
Sigma Vth
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin


Primary Source of Variation
Vth Variation
Linear Change in Ron
Exponential Change in Roff
Roff variation dominates
all other physical variation
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin


Preventing Overlap
Ron (Vin=Vdd)
Roff (Vin=0)
Height = 1e+2
R (Ohm)
Sigma Vth
-How to avoid overlap problem?
-Mark leakiest defective
-Perform Variation Oblivious mapping
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin


Defect-Avoiding Algorithm
-Mark leakiest resources as defective
-Perform Variation-Oblivious mapping on remaining resources
SPLA at 38% Variation
TLeak
100×TSwitch
NAND-Term Count
Time (s, log scale)
Works but discards 48% of resources, 167% wider channels leading to high delay, energy and area
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin


Logical Variation: Variation in Fanout
τLeak ≈ Roff · ΣFanout(Cout)
Fanout comes from netlist
Map:
AB+ACD+BE+AF
A B C D E F
AB
ACD
BE
AF
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Logical Variation: Variation in Fanout

$\tau_{Leak} \approx R_{off} \cdot \Sigma_{Fanout}(C_{out})$

A has fanout 3:
AB, ACD, AF

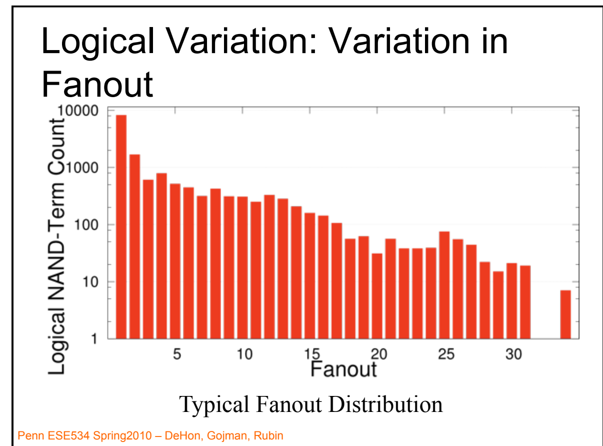
F has fanout 1:
AF

Even at nominal, A sees 3 times more load than F

A B C D E F
AB
ACD
BE
AF

Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Logical Variation: Variation in Fanout



Typical Fanout Distribution

Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Defect-Avoiding Toy Example

Step 1 Discard Leaky Resources

Logical Fanout	Roff Resistance	Estimated τ_{Leak}
1	320	
10	20 (discarded)	
15	150	
1	300	
3	100	
	30	

Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

Defect-Avoiding Toy Example

Step 2 Match Functions to Resources

Logical Fanout	Roff Resistance	Estimated τ_{Leak}
1 →	320	320
10 ↘	20 (discarded)	
15 ↘	150	1500
1 ↘	300	4500
3 ↘	100	100
	30	900

Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

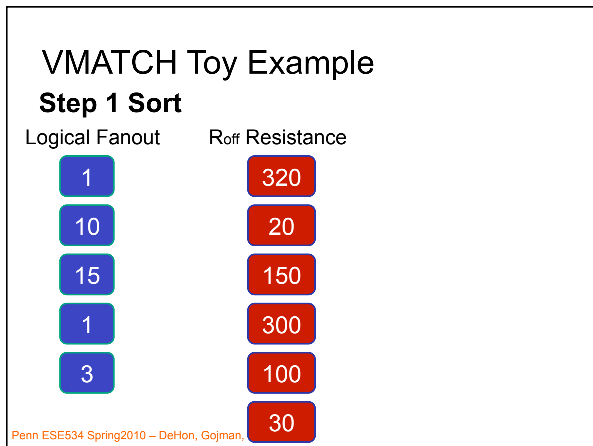
VMATCH

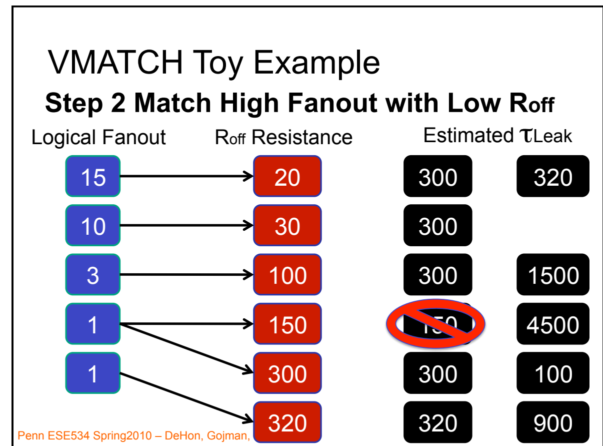
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

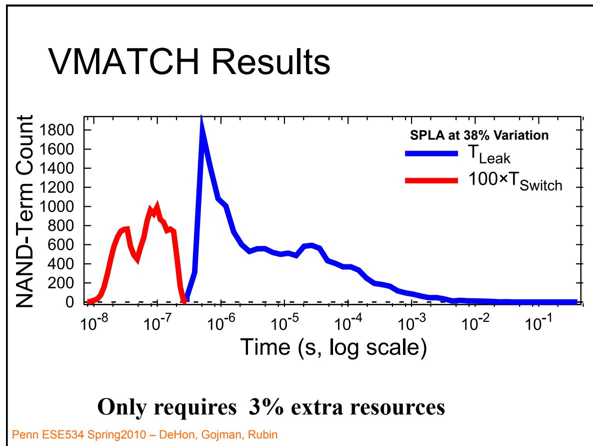
VMATCH

- Logical Variation + Physical Variation >> Variation
- Variation-Aware Mapping Algorithm
- Uses logical variation (fanout variation) to counteract physical variation (Roff variation)
  - Matches high-fanout term with low Roff NAND-Term and vice versa.

Penn ESE534 Spring2010 – DeHon, Gojman, Rubin







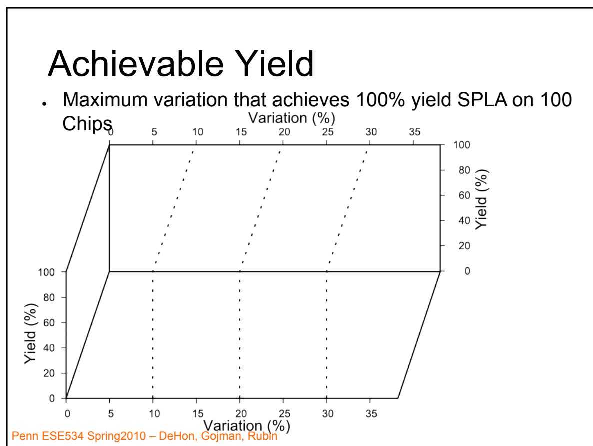
Experiments

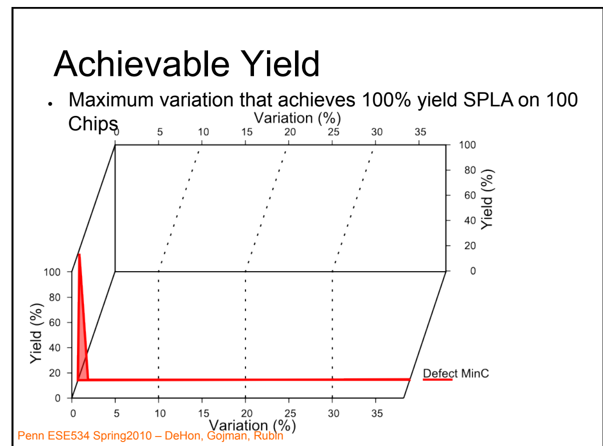
5nm

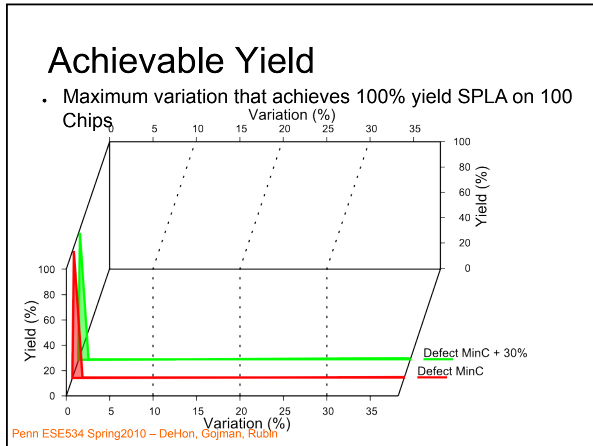
5nm

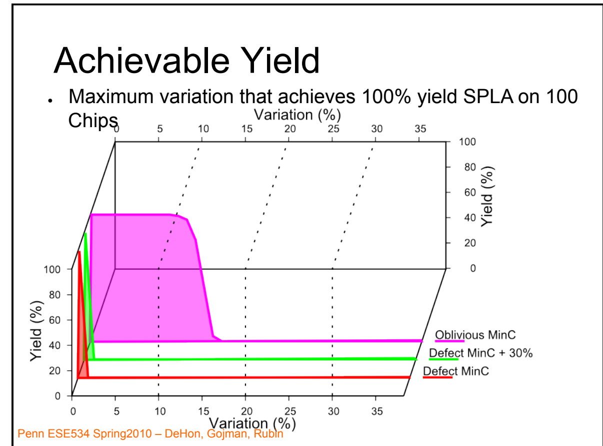
- Simulated NanoPLA using 5nm pitch wires, 5nm transistor channel length
- Modeled variation up to ITRS predicted 38% for technology
- Implemented Variation-Oblivious, Defect-Avoiding and VMATCH Algorithms on NPR
- Routed Toronto 20 Benchmark on 100 Monte Carlo generated chips.

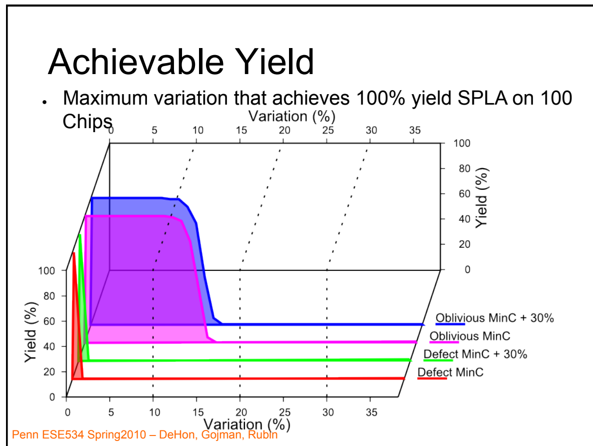
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

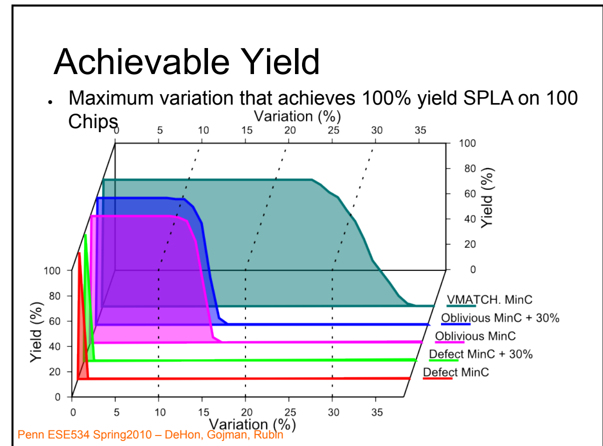


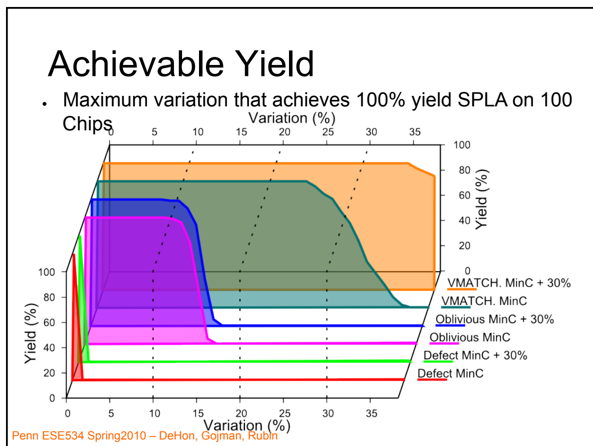












Matching Lesson 2

- Can keep yield high
  - @ 38% variation, small amount of extra wires
- …despite the fact that Roff varies over 12 orders of magnitude
- Trick is to match the logical variation to the physical variation to reduce both

78

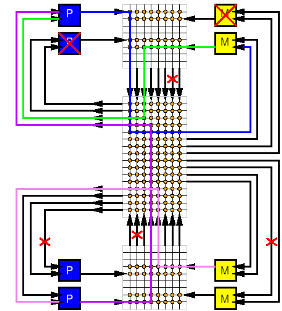
Penn ESE534 Spring2010 – DeHon, Gojman, Rubin

FPGA Defects

Full Knowledge

Problem

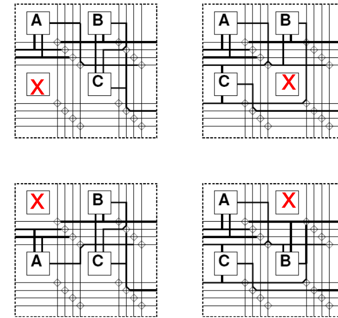
- At high defect rates, some resources will be unusable
- Resources
  - LUTs
  - FFs
  - Interconnection links
- Full crossbars are too expensive



Opportunity

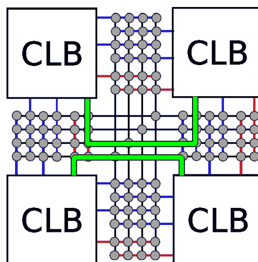
- Avoid defective elements during mapping
  - Don't place logic in defective LUT
    - Like disk-drive model where we don't allocate a defective sector to hold data
  - Route to avoid defective interconnect

Avoid Defective LUT



[Lach *et al.* / FPGA 1998]

Avoid Defective Interconnect



Example

- Collect 4 numbers between 1 and 81

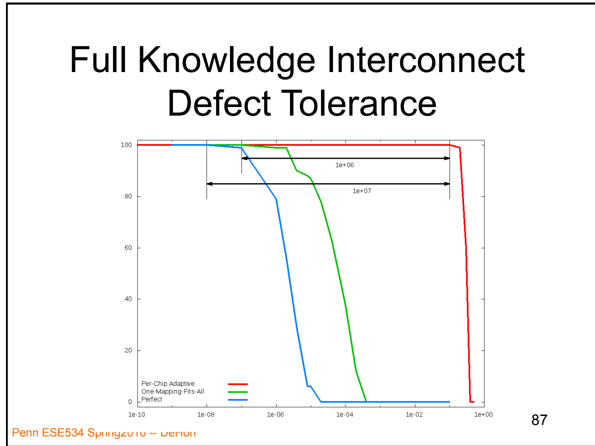
Defects on Mesh

- Draw on board for mapping on next slide.

Class Exercise

- Map this circuit to the defective LUT mesh, avoiding defects.

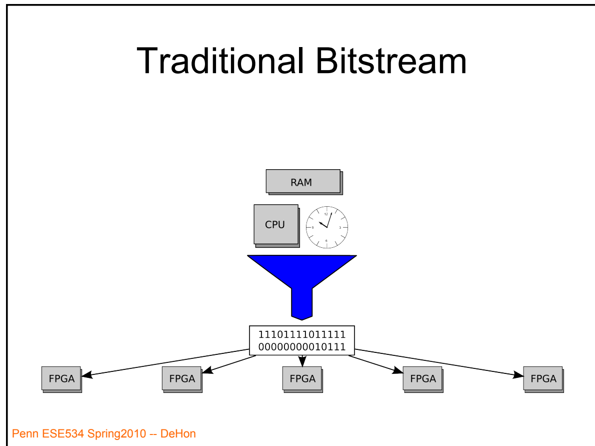
Full Knowledge Interconnect Defect Tolerance



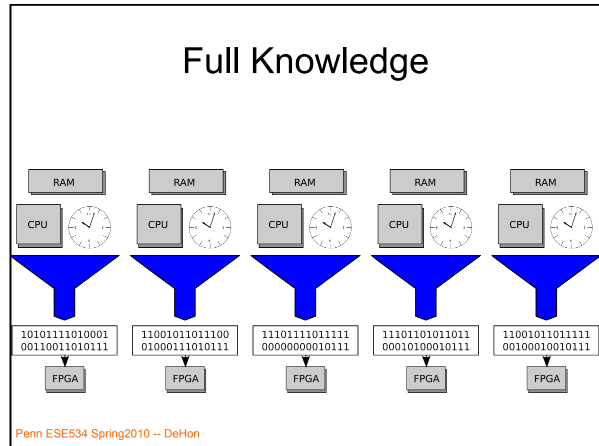
FPGA Routing Defects

Pre-computed Alternatives
Lightweight Defect Tolerance
Choose Your own Adventure

Traditional Bitstream



Full Knowledge



Choose Your own Adventure

- Bitstream with alternatives
  - Still just one bitstream

Choose Your own Adventure

- Bitstream with alternatives
  - Still just one

Loading

Examine path
- Configure
- Test
  - Good: skip to next net
  - Bad: grab next path

Loading

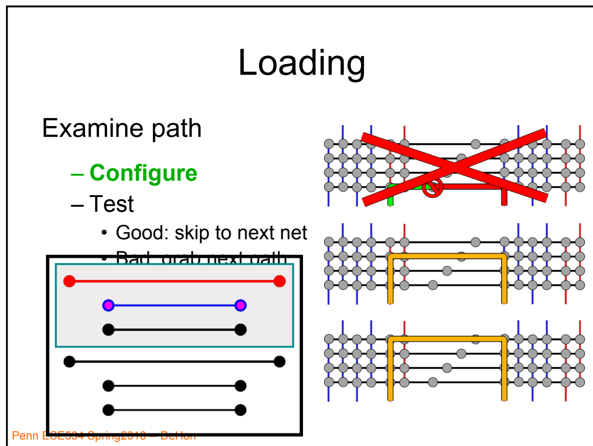
Examine path
- **Configure**
- Test
  - Good: skip to next net
  - Bad: grab next path

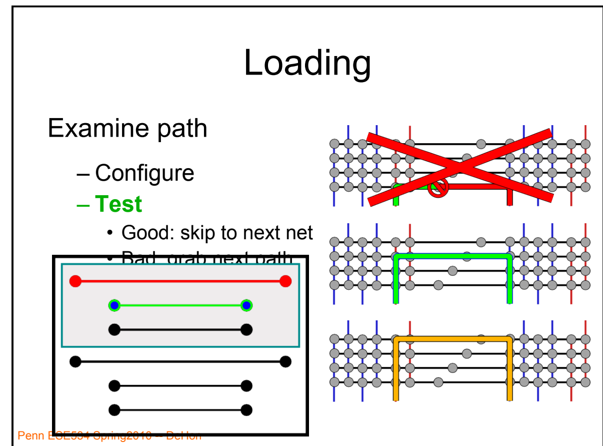
Loading

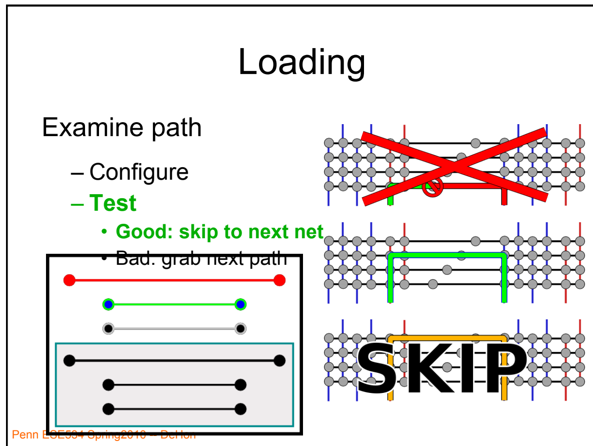
Examine path
- Configure
- **Test**
  - Good: skip to next net
  - Bad: grab next path

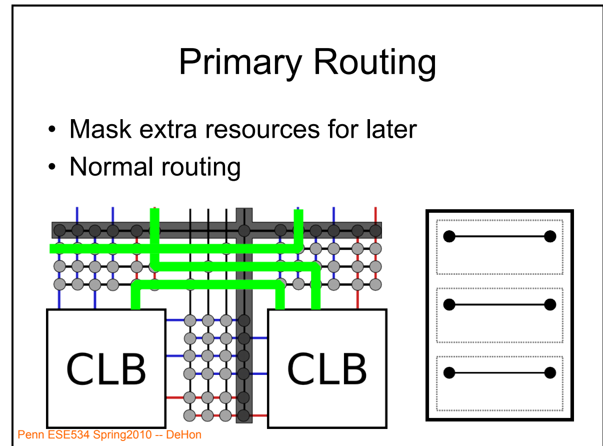
Loading

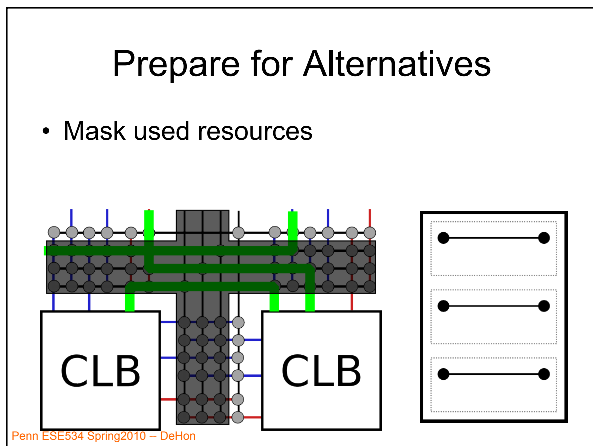
Examine path
- Configure
- **Test**
  - Good: skip to next net
  - **Bad: grab next path**

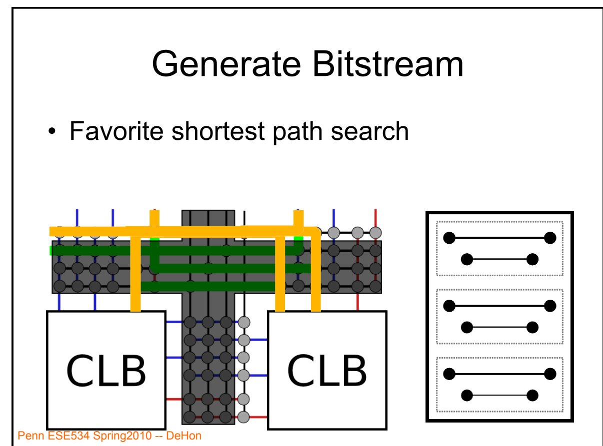

Loading
Examine path
– Configure
– Test
• Good: skip to next net
• Bad: grab next path
Penn ESE534 Spring2010 -- DeHon

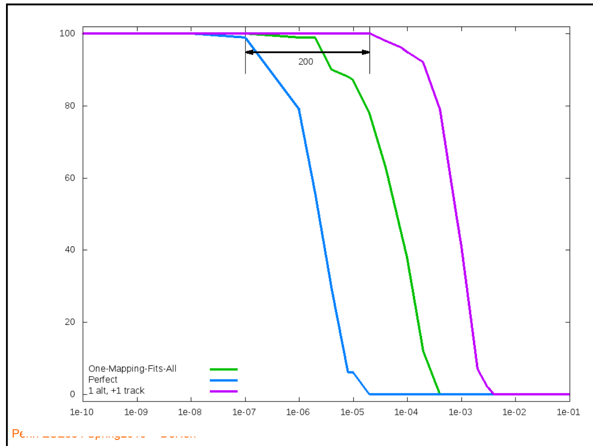

Loading
Examine path
– Configure
– Test
• Good: skip to next net
• Bad: grab next path
Penn ESE534 Spring2010 -- DeHon

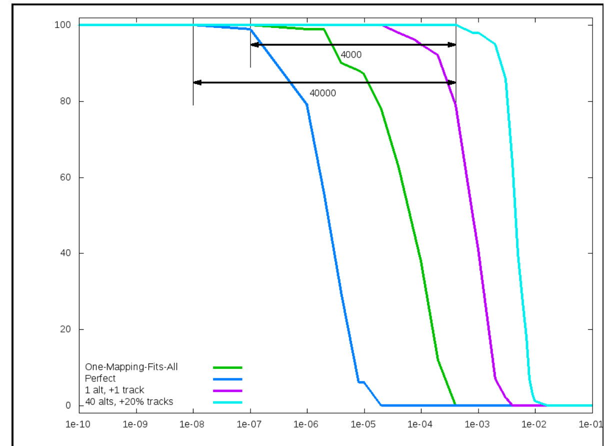

Loading
Examine path
– Configure
– Test
• Good: skip to next net
• Bad: grab next path
SKIP
Penn ESE534 Spring2010 -- DeHon

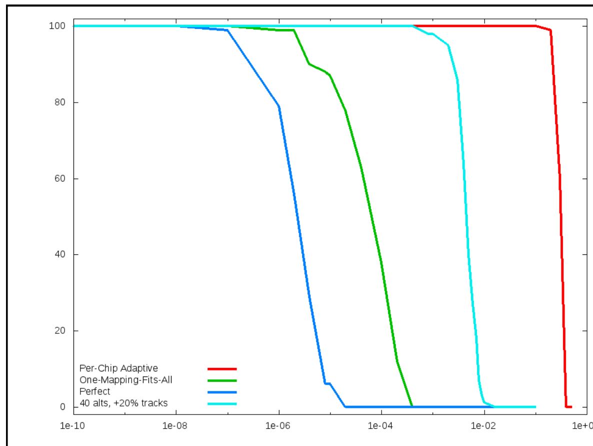

Primary Routing
• Mask extra resources for later
• Normal routing
CLB
CLB
Penn ESE534 Spring2010 -- DeHon


Prepare for Alternatives
• Mask used resources
CLB
CLB
Penn ESE534 Spring2010 -- DeHon


Generate Bitstream
• Favorite shortest path search
CLB
CLB
Penn ESE534 Spring2010 -- DeHon







Admin

- Homework 7 due Monday
- No office hours today (André away)
- Reading for Monday on web
  - Classic paper on Rent's Rule

Big Ideas [MSB Ideas]

- Nanoscale Integrated Circuits are like snowflakes – each unique
- Cannot expect perfect ICs
  - …or even perfect building blocks
- But even defective components are often "good enough"
- Post-fabrication matching allows us to assign functions based on capabilities

Big Ideas [MSB-1 Ideas]

- Matching in nanoPLA
  - 5% crosspoint non-programmable defects
  - σ=38% variation in V_{th}
- Defect-aware routing in FPGAs
  - 10% switch and segment defects with full knowledge
  - 1% with pre-computed alternatives