Performance-Aware CMP Fill Optimization

Andrew B Kahng[1,2] and Rasit Onur Topaloglu[1,3]

University of California at San Diego
Computer Science and Engineering Department[1]
Electrical and Computer Engineering Department[2]
Advanced Micro Devices[3]

abk@cs.ucsd.edu rtopalog@cs.ucsd.edu rasit.topaloglu@amd.com

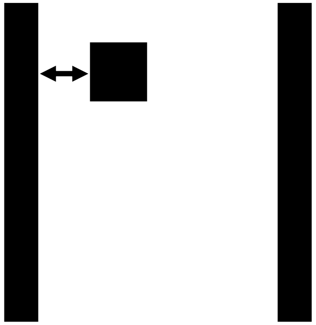
Invited Paper, VMIC, 2007

Outline

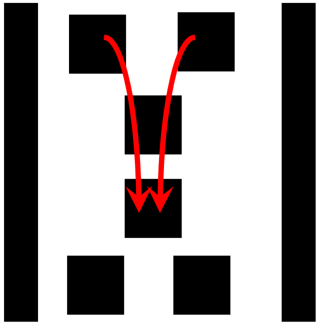
- Introduction
- Traditional Fill Insertion Methodology
- Proposed Fill Insertion Methodology
  - Adaptive Regions
  - Grid Model Utilizing Bonds
  - Energy Modeling for Bonds
- Experimental Setup and Protocol
  - Flow for Fill Insertion
  - Timing, Density Histograms and Pattern Comparison
- Conclusions and Ongoing Work

Introduction

- CMP fills inserted to make metal density uniform
  - Reduces post-polish height variations
  - Improves electrical variability
- Classical methods focus on density uniformity only
  - Impact on circuit performance minimally considered
- We develop a heuristic method, which
  - minimizes coupling capacitances
  - analogous to electrons filling energy levels in an atom
  - is able to utilize known fill insertion guidelines



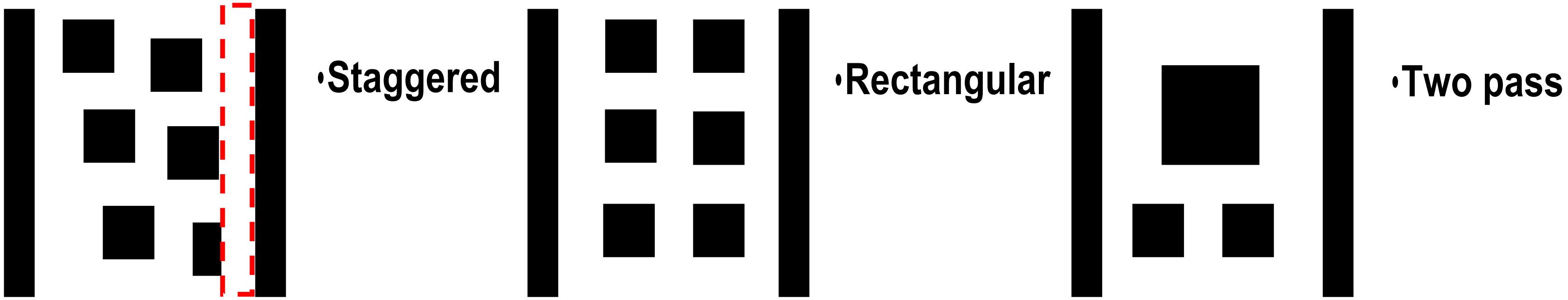
- Place fills away from interconnects
- Provides flux path towards other interconnects as distance increases



- Place fills to form a hour-glass shape
- A path to facilitate parallel flux flow is generated
- Minimal number of fills close to interconnects

Traditional Fill Insertion

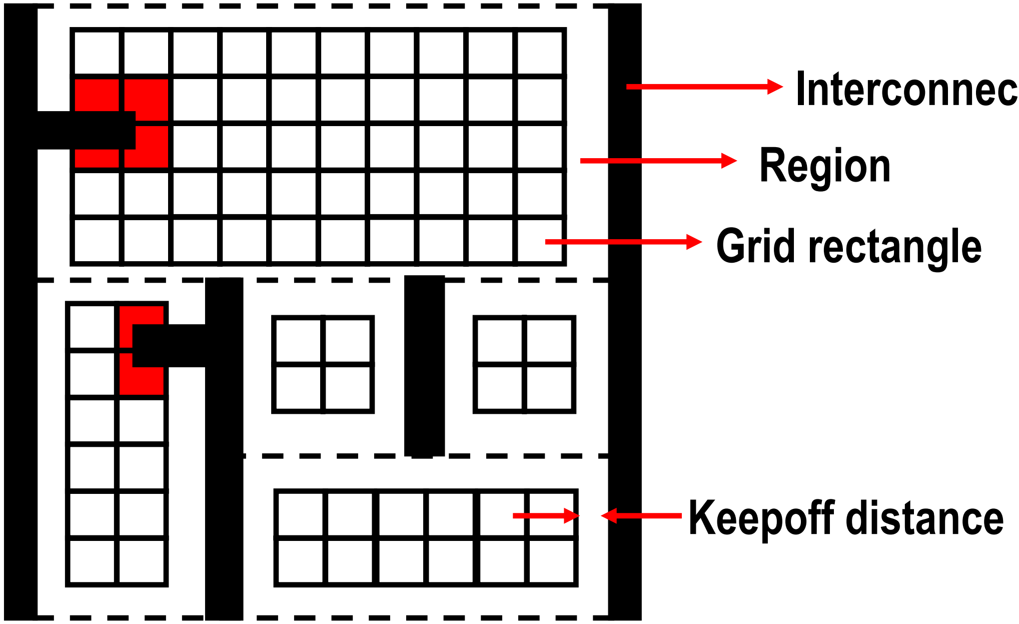
- Fills may have a particular pattern



- Keepoff (exclusion) rules used to define additional space between a fill and an interconnect
- Layout filled stepping through fixed size dissections or windows
- Fills are inserted as long as there is space
  - Fill size and spacing adjusted to yield a target density
- Improvements exist that consider minimum density variation across windows, limit number of fill shapes

Adaptive Region Definition

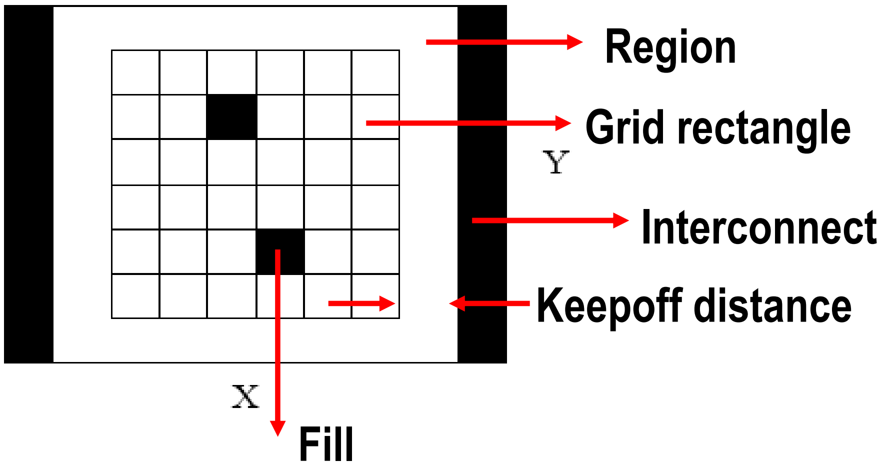
- We use region-based fill insertion
- Maximum width empty regions identified between facing interconnects using our scanline algorithm



- After stripping out keepoff distances, a grid holding possible fill locations is formed
- If orthogonal interconnect segments exist, overlapping grid rectangle locations are disabled

The Grid Model Utilizing Bonds

- There are 36 rectangles with two fills in the grid shown below



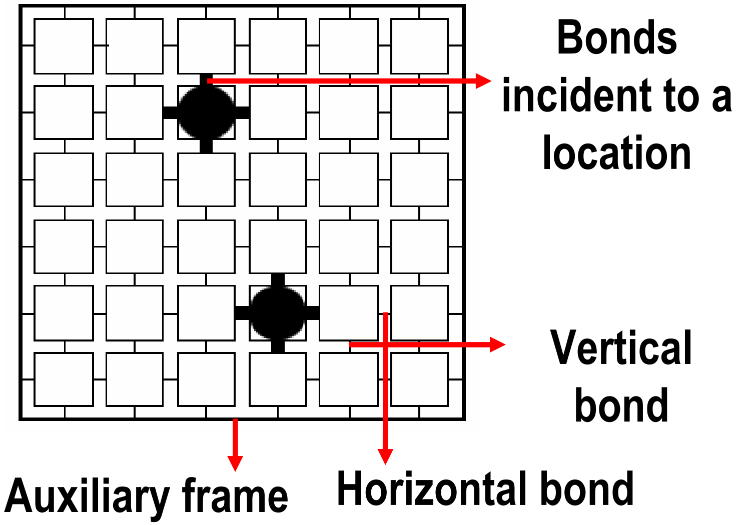
Region

Grid rectangle

Interconnect

Keepoff distance

Fill



Bonds incident to a location

Vertical bond

Auxiliary frame

Horizontal bond

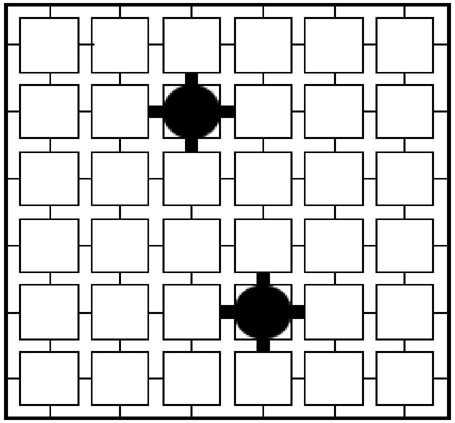
- An auxiliary frame is formed holding grid rectangles with bonds in between
- Each bond has an adjustable energy
- When inserting a fill, bonds incident to a rectangle are summed up to find an energy; we find minimum energy location to insert a fill

Energy Modeling in a Grid

- Modeling of bonds indicates which location should be filled with higher priority
- Model is flexible enough to satisfy target guidelines
- Adjustable four-parameter model for vertical and horizontal bonds
  - Although we use linear models, second-order and more complex models can also be used

Vertical model: $\gamma + (|(i_{mid}) - i| * \zeta / (i_{mid}))$

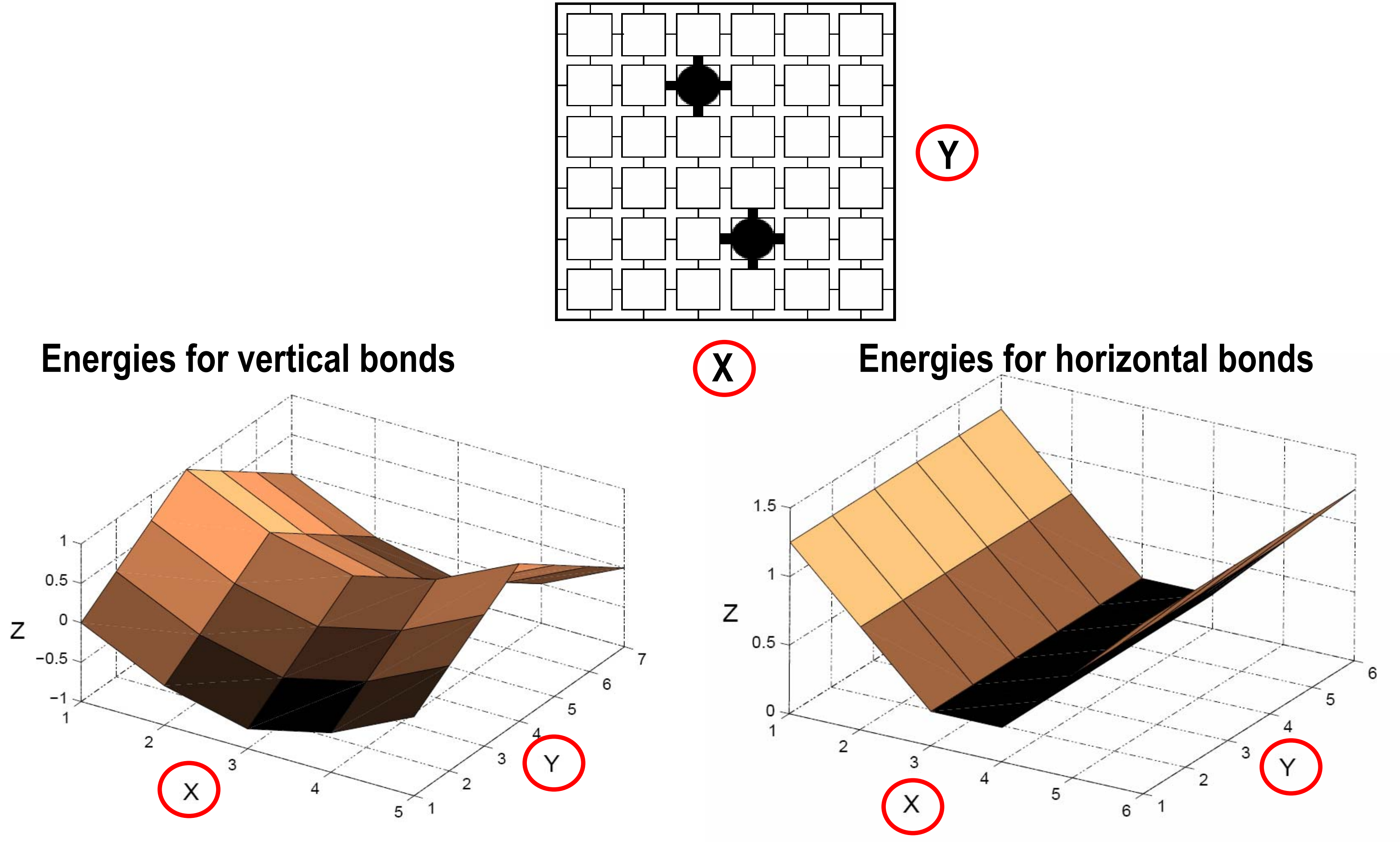
Horizontal model: $\alpha + (|(j_{mid}) - j| * \beta / (j_{mid}))$



i : Enumeration for a row of grid rectangle locations
j : Enumeration for a column of grid rectangle locations
i_{mid} : Middle row number
j_{mid} : Middle column number
$\alpha,\beta,\gamma,\zeta$: Fitting parameters

Example of Bond Energies in a Grid

- Below, we show an example of bond energies
- Z axis gives the bond energy



Energies for vertical bonds

Energies for horizontal bonds

Fill Size and Spacing Selection

$$D = (W \cdot A_r - A_i)/A_r$$

$$FW = (1 + \varepsilon)\sqrt{D}$$

$$FS = \sqrt{1 - D}$$

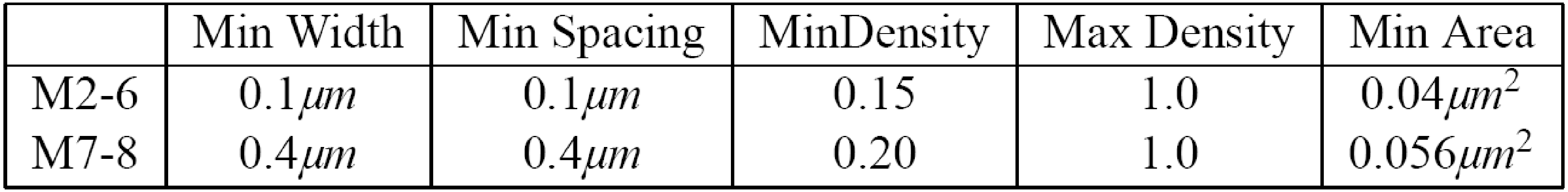
D : Target fill density
W : Target feature density
A_r : Area of region
A_i : Area of interconnects in A_r
FW : A parameter proportional to fill width
FS : A parameter proportional to fill spacing
Epsilon (ε) : Parameter used for space adjustment during optimization

- We iteratively scale up both FW and FS by *s* until design rules, i.e., fill width, spacing and area, are satisfied
  - Scale down if bounds exceeded by smaller constant, or alternatively, fix fill width or spacing to given constants first
- Grid rectangle size is selected as $s^n * (FW + FS)$, where *n* is the number at which we cut off scaling
- Larger ε provides increased space for optimization
- We set fill width and spacing ratio proportional to FW/FS and set ε 0 for traditional fill

Experimental Setup and Protocol

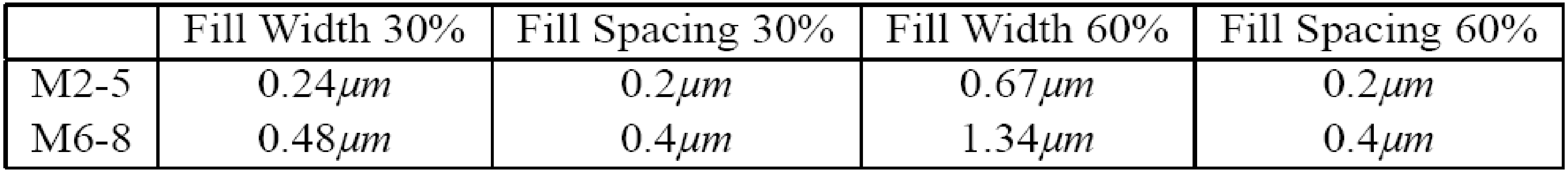
- Cadence SOC Encounter v5.2 used for placement and clock tree synthesis and NanoRoute used for routing
- Synopsys StarRCXT 2006.06 used for RC extraction
- Calibre 20071_34 used for fill insertion
- We use C++ code for proposed fill insertion methodology MFO (Metal Fill Optimizer)
- We use TSMC 65nm GPlus library

Fill Design Rules from Library Exchange File



	Min Width	Min Spacing	MinDensity	Max Density	Min Area
M2-6	$0.1\mu m$	$0.1\mu m$	0.15	1.0	$0.04\mu m^2$
M7-8	$0.4\mu m$	$0.4\mu m$	0.20	1.0	$0.056\mu m^2$

Sizes for Traditional Fill



	Fill Width 30%	Fill Spacing 30%	Fill Width 60%	Fill Spacing 60%
M2-5	$0.24\mu m$	$0.2\mu m$	$0.67\mu m$	$0.2\mu m$
M6-8	$0.48\mu m$	$0.4\mu m$	$1.34\mu m$	$0.4\mu m$

Flow for Fill Insertion

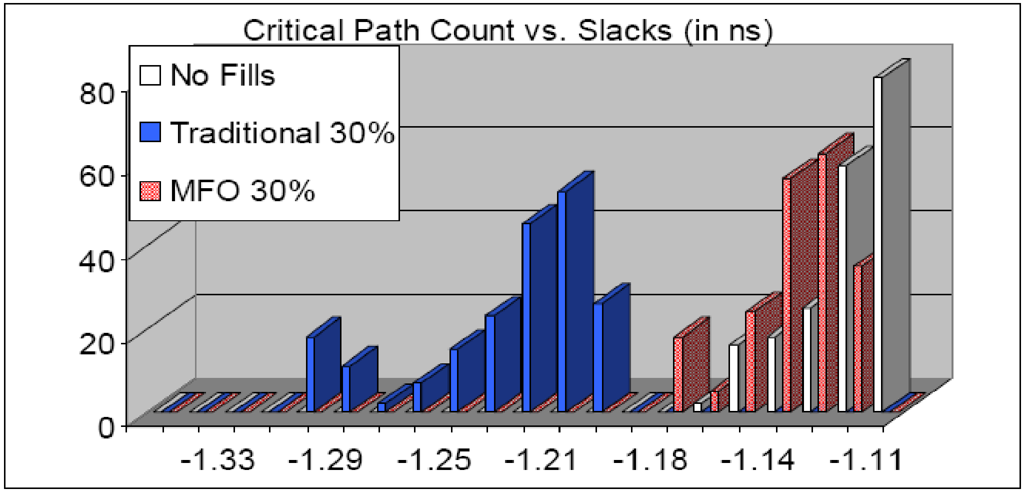
1. Place, synthesize clock network and route design
2. Using GDS from Step 1, insert fills into GDS
3. Use Perl scripts to obtain DEF containing fills
4. Extract .spef parasitics from DEF containing fills
5. Run static timing analysis using .spef file from Step 4
6. Use Calibre scripts to obtain density histograms
7. Use Perl scripts to obtain slack distribution

- For no fill case, we bypass step 2
- Input can be GDS or DEF
- We use $\alpha=0$ $\beta=1$ $\gamma=0$ $\zeta=-1$ $\varepsilon=0.5$ for optimization parameters

Testcases

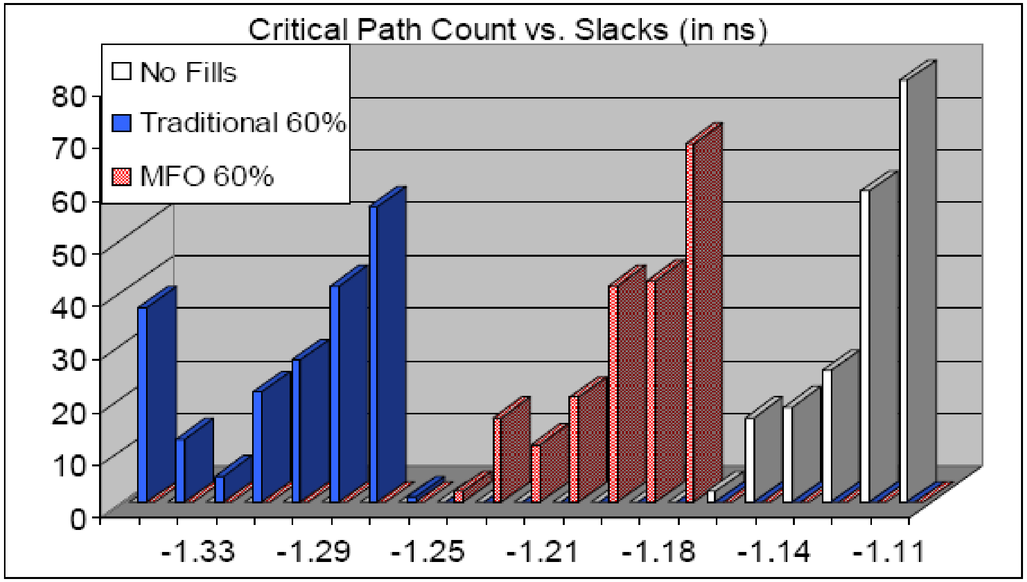
- s38417 circuit
  - 8 metal layers, 8514 gates, 43,076 interconnect segments
- Compare 30% target metal density between no fill, traditional and MFO optimized
- Compare 60% target metal density between no fill, traditional and MFO optimized
- Compare 30% target metal density between traditional, traditional with 2x the fill width and MFO optimized
- Compare 30% target metal density between traditional, MFO optimized with epsilon 0.5 and 0

Discussion of Results



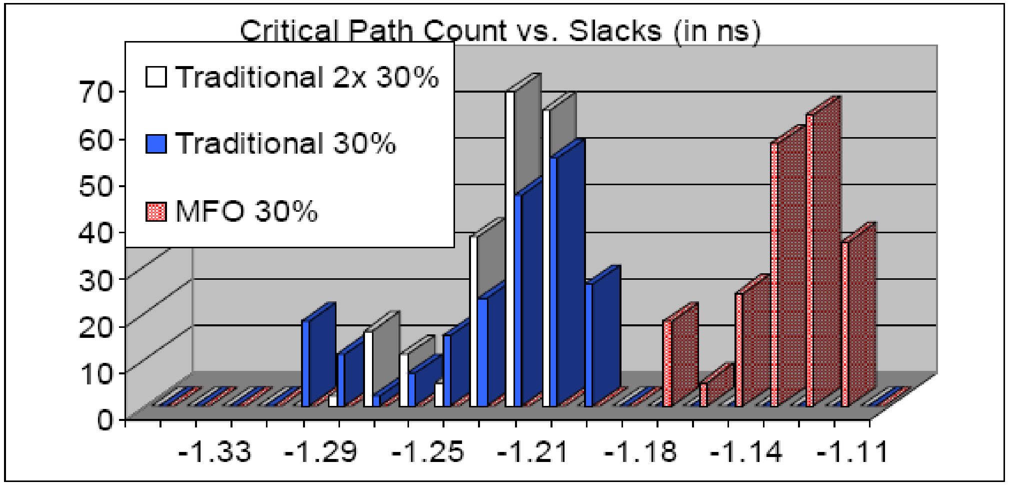
- Timing slacks shown
- Less negative is better

- For 30% density, MFO almost similar to no fills

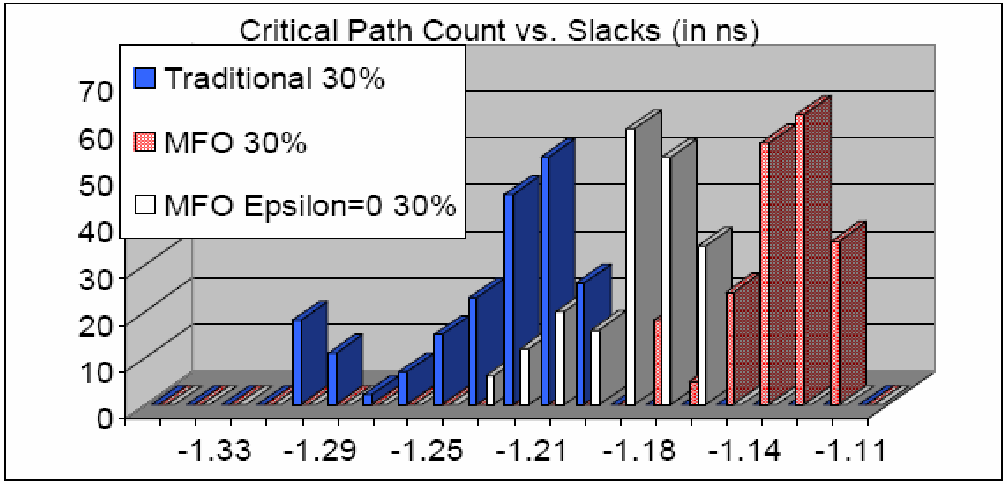


- For 60%, MFO still outperforms traditional fill

Impact of Fill Size and Epsilon



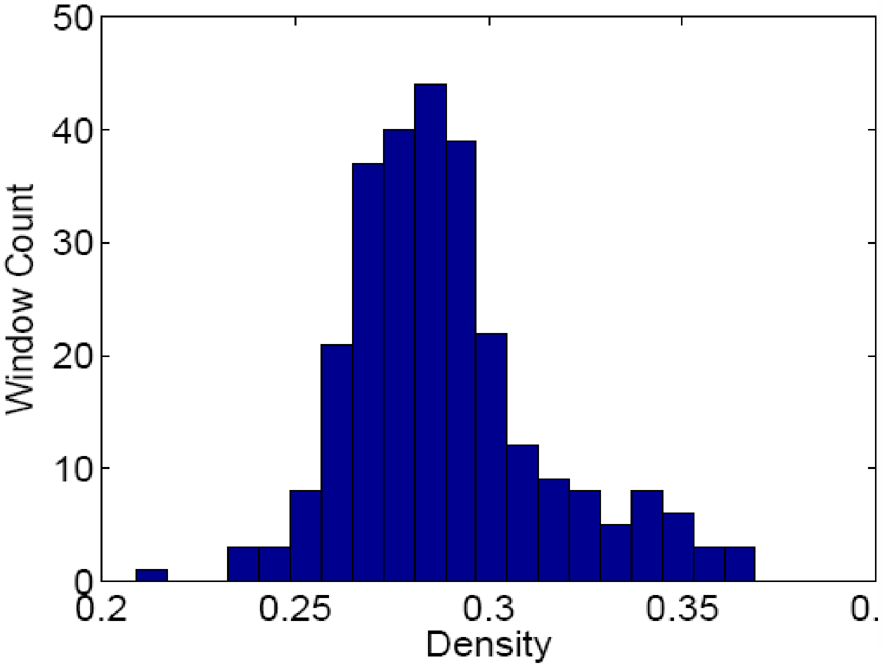
- Increasing fill size by 2x does not bring improvement, though number of fills decreases to 1/4th

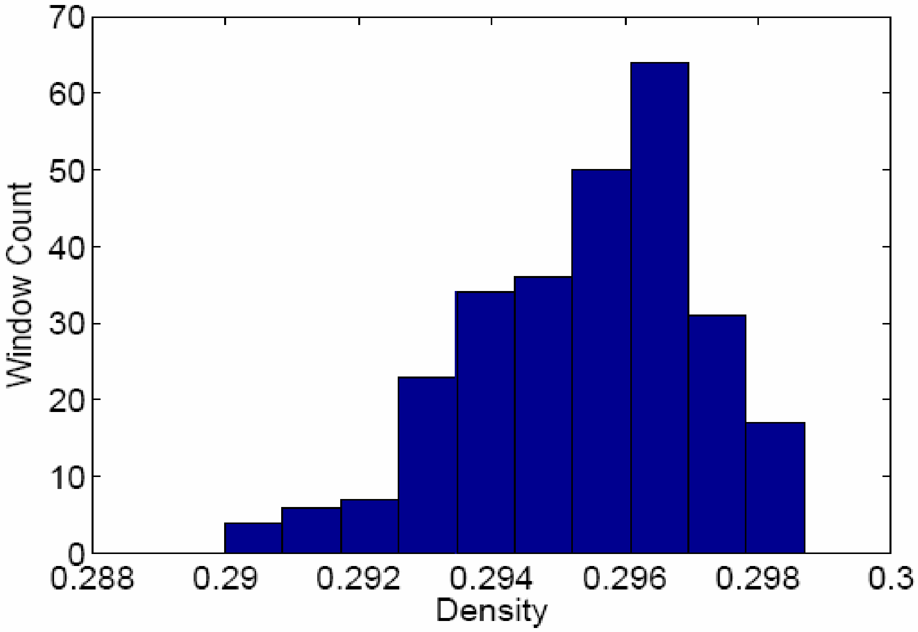


- Decreasing the epsilon parameter results in less space for optimization, hence degrades results

Density Histograms

- For 30% target density,
  - MFO results in 28.87% mean density with σ = 2.55%
  - Traditional fill results in μ = 29.54% with σ = 0.002%
- Chip-level structure (scribe line and module to module density differences) dominates the density variation, hence the loss in window density variation is acceptable

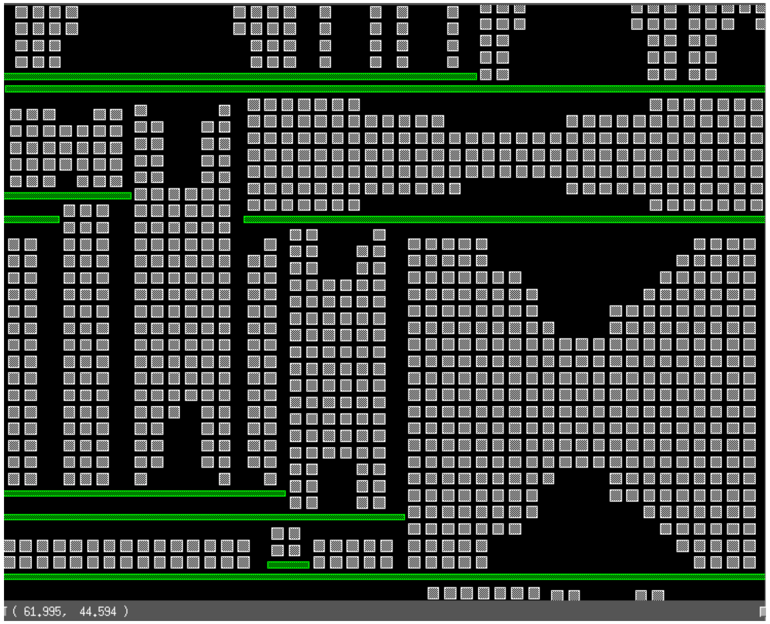




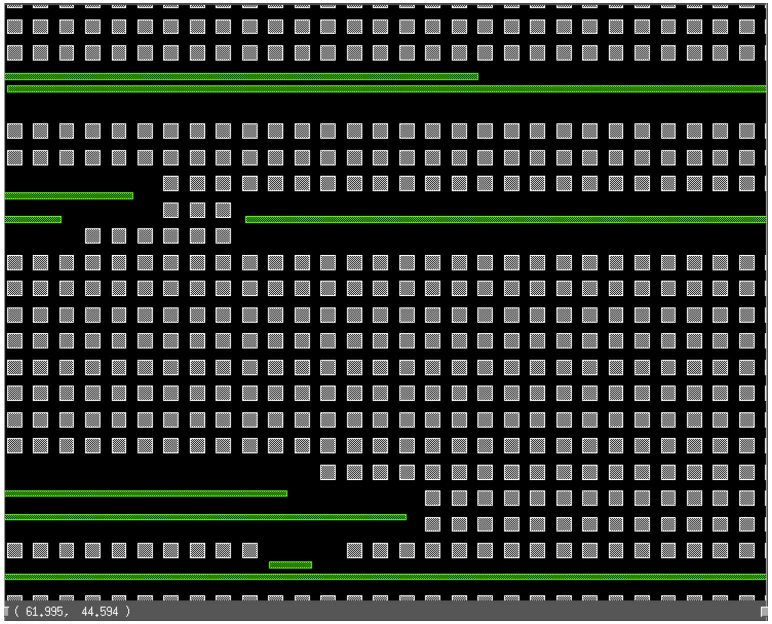
Fill Pattern Comparison – MFO vs Traditional 30%

- Same area filled using two methods shown
- We achieve the target hourglass shape using MFO

MFO 30%



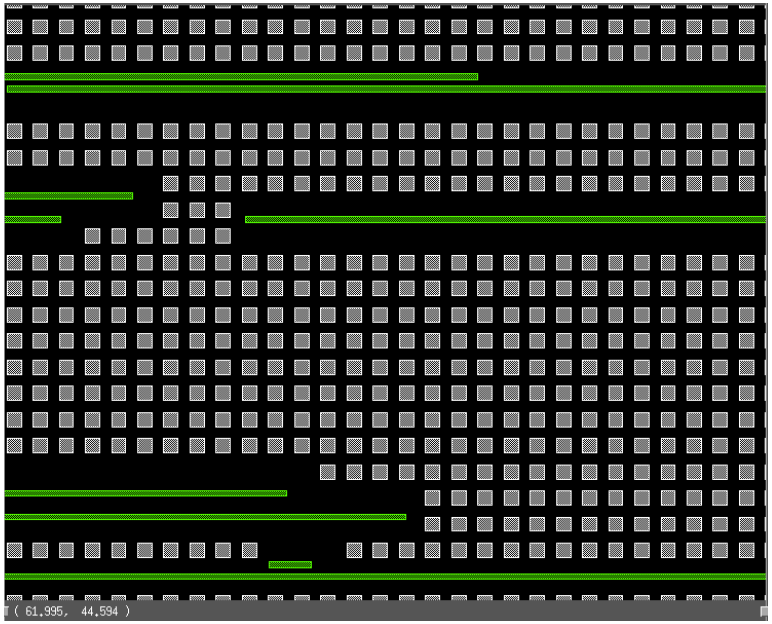
Traditional 30%



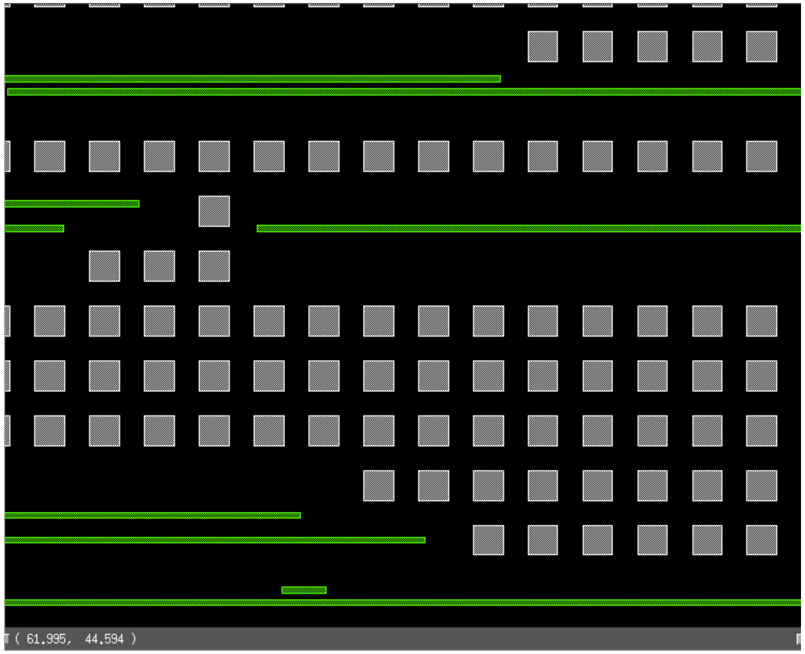
Fill Pattern Comparison – Traditional vs Traditional 2x

- Increasing fill width by 2x reduces number of fills to 1/4th
- This has negligibly improved timing

Traditional 30%



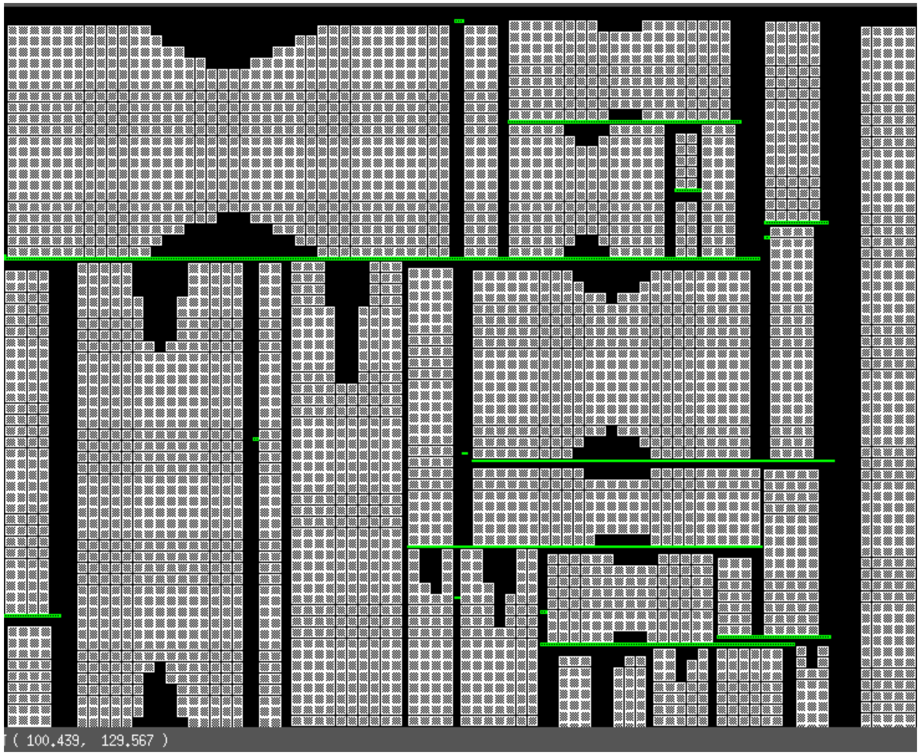
Traditional 30% with 2x Fill Width



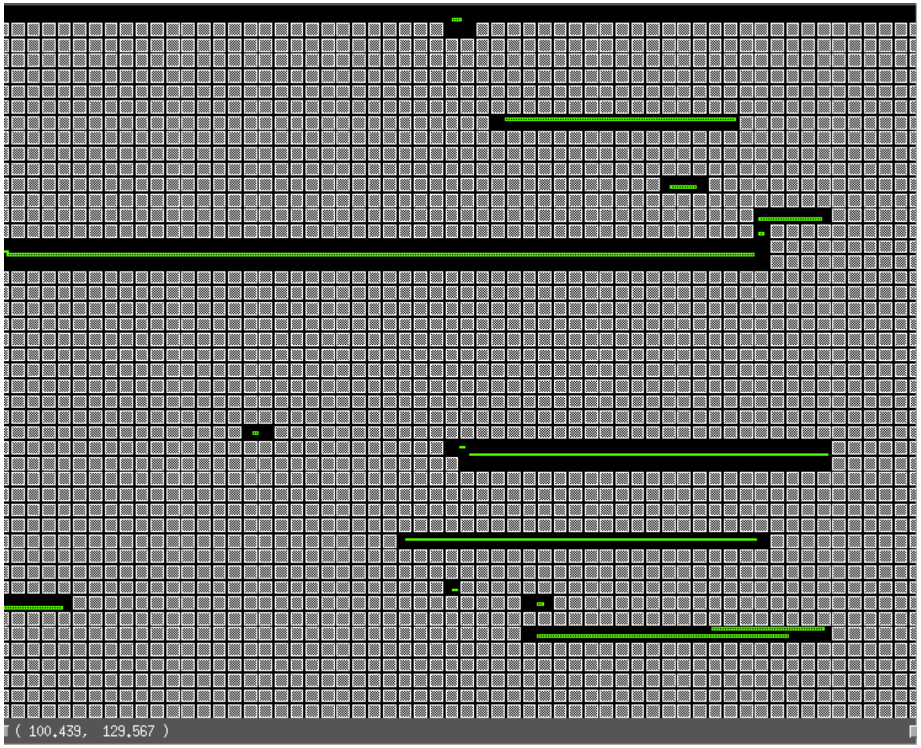
Fill Pattern Comparison – MFO vs Traditional 60%

- Same area filled using two methods shown for 60%

MFO 60%



Traditional 60%



Conclusions

- Physically-motivated heuristic
- Testbed with 65nm Gplus process and fill design rules
- It is possible to reduce the fill impact on timing
  - by up to 85% for 30% pattern density
  - by up to 65% for 60% pattern density

Ongoing Work

- Couple optimization with new 45nm AMD CMP models
- Critical-net aware fill optimization
- Inter-layer aware fill optimization