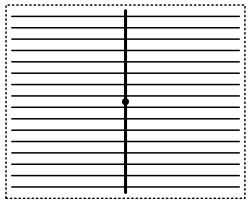
Stratix™10 High Performance Routable Clock Networks

Carl Ebeling, Dana How, Herman Schmit, David Lewis

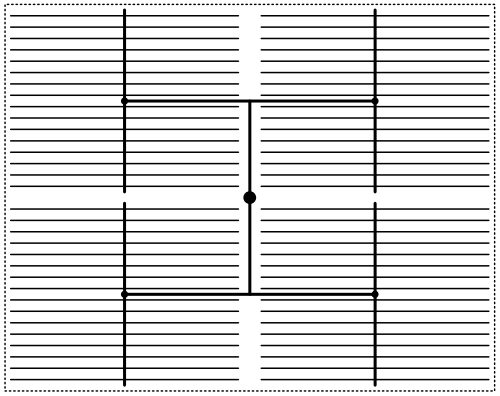
In the beginning, there was an FPGA

- and clocks were simple

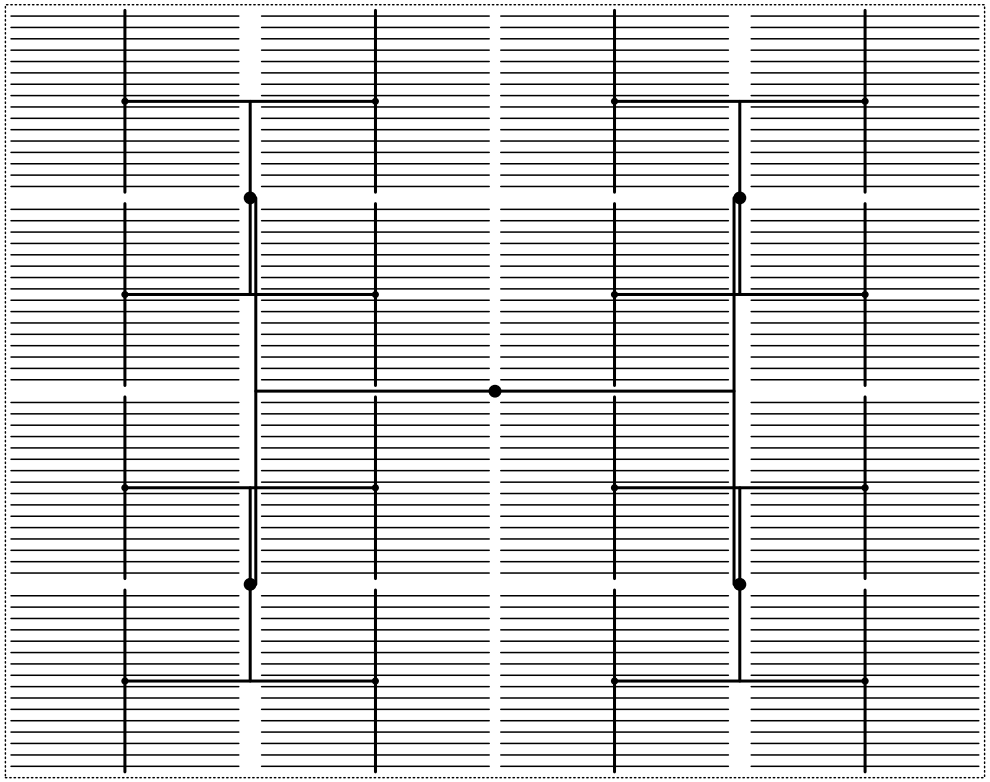


But Moore's Law was relentless

- FPGAs became larger
- We needed balanced trees

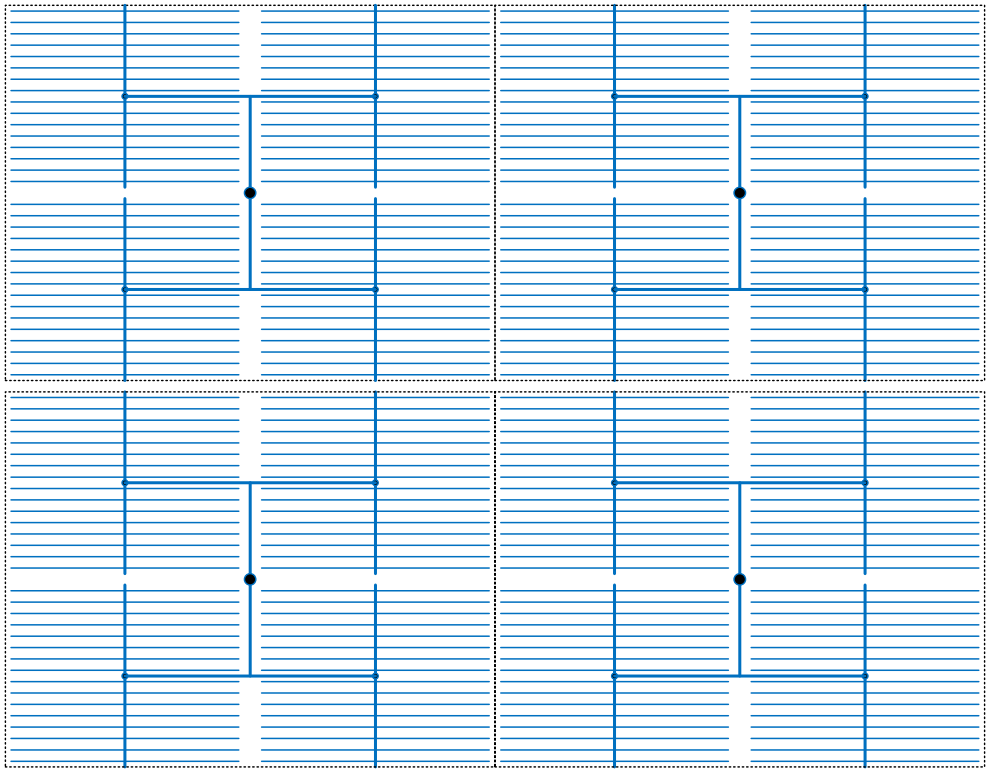


Clock Trees Became Even Larger

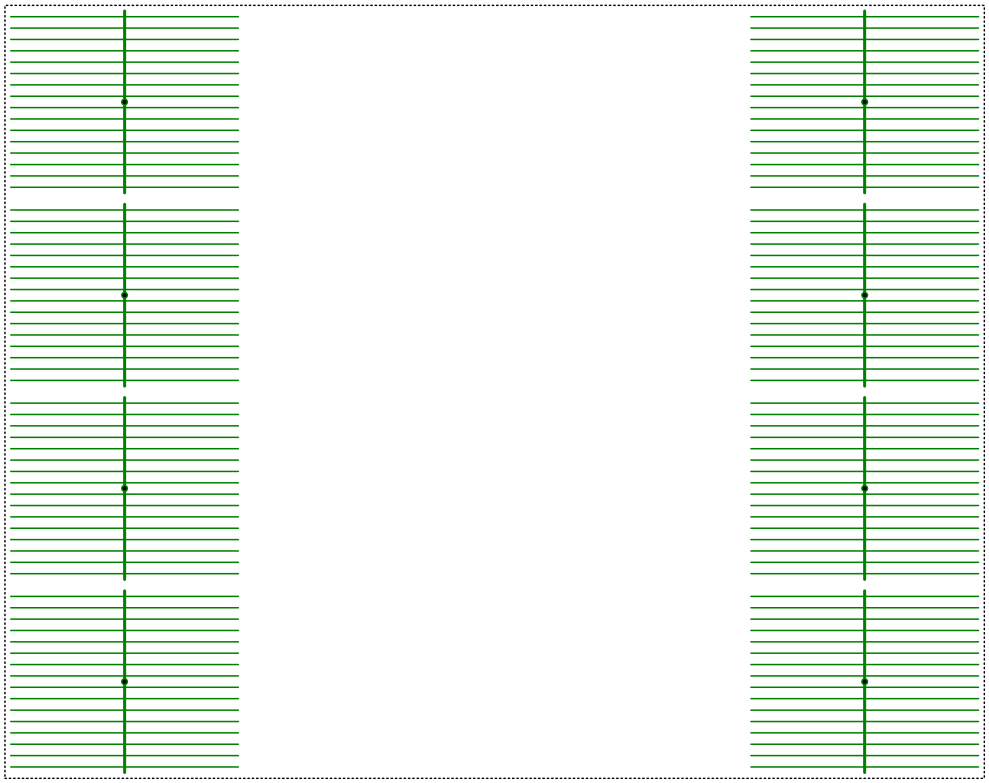


Different Sizes of Fixed Clock Trees

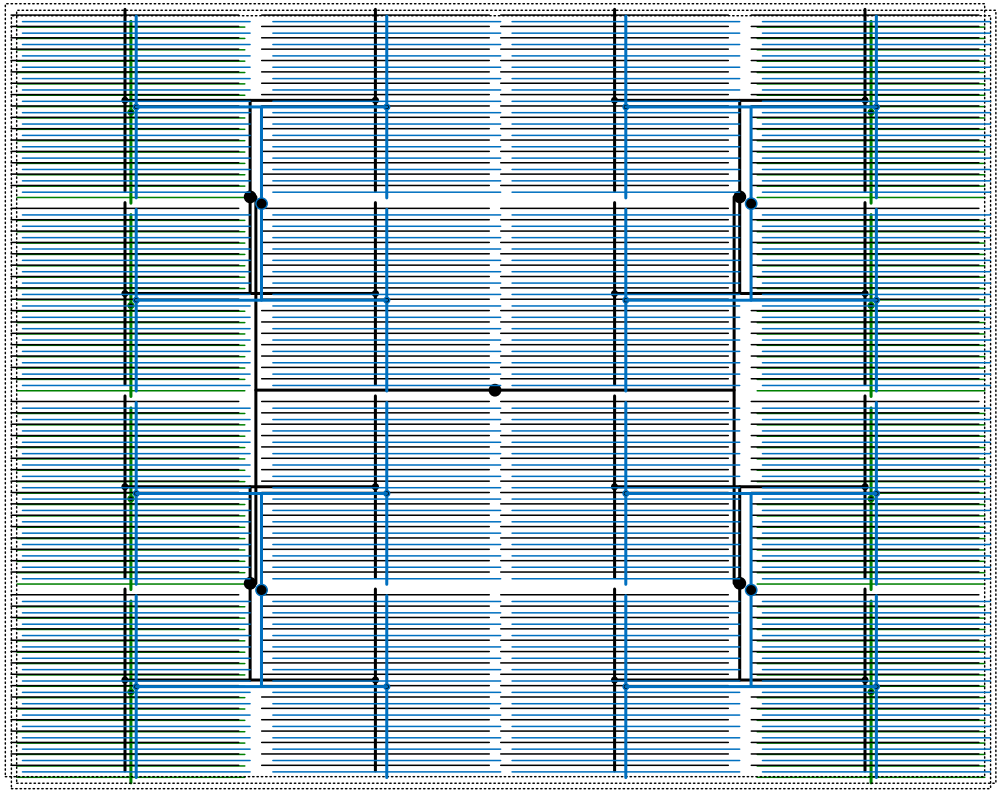
- Quadrant Clocks



Peripheral clocks



Overlay them all

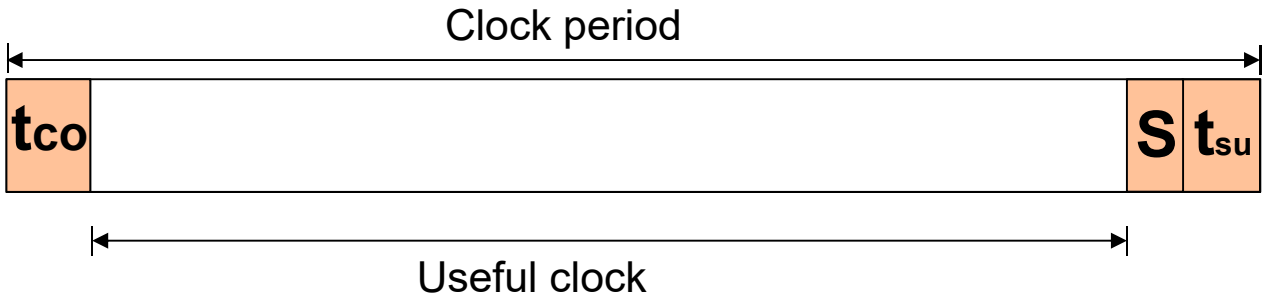


This Approach Does Not Scale Well

- Cost:
  - How many fixed clocks?
  - What size trees?
- Performance

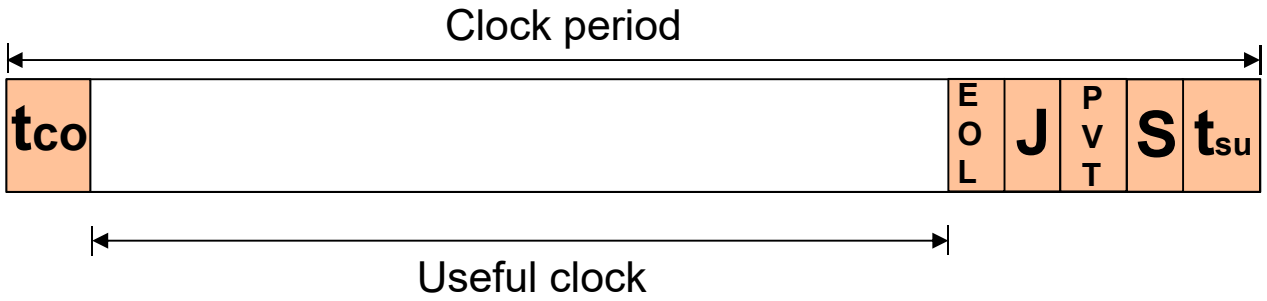
Clock Distribution and Clock Loss

- Register setup (t_{su}) and propagation delay (t_{co})
- Clock skew (S)



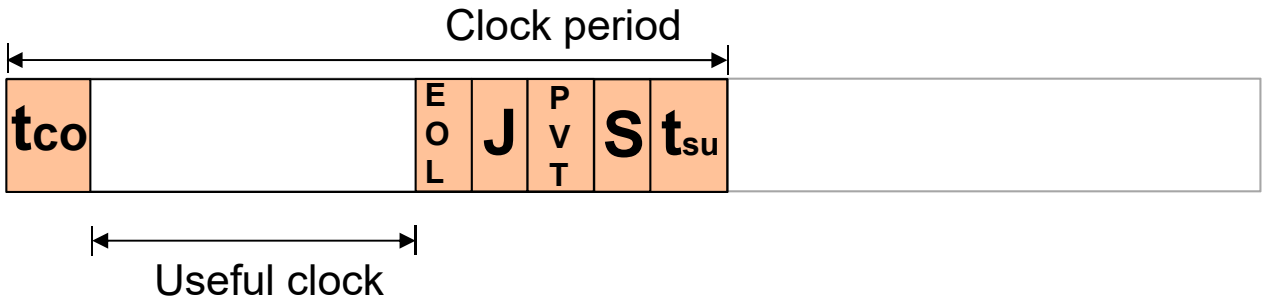
The Performance Implications of Clock Distribution

- **Advanced technologies**
  - Variation: Process, voltage, temperature
  - Jitter
  - Aging/End of Life



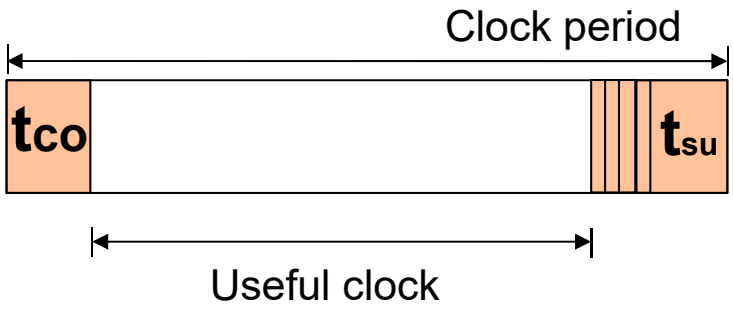
The Performance Implications of Clock Distribution

- And now we want to double the clock frequency!



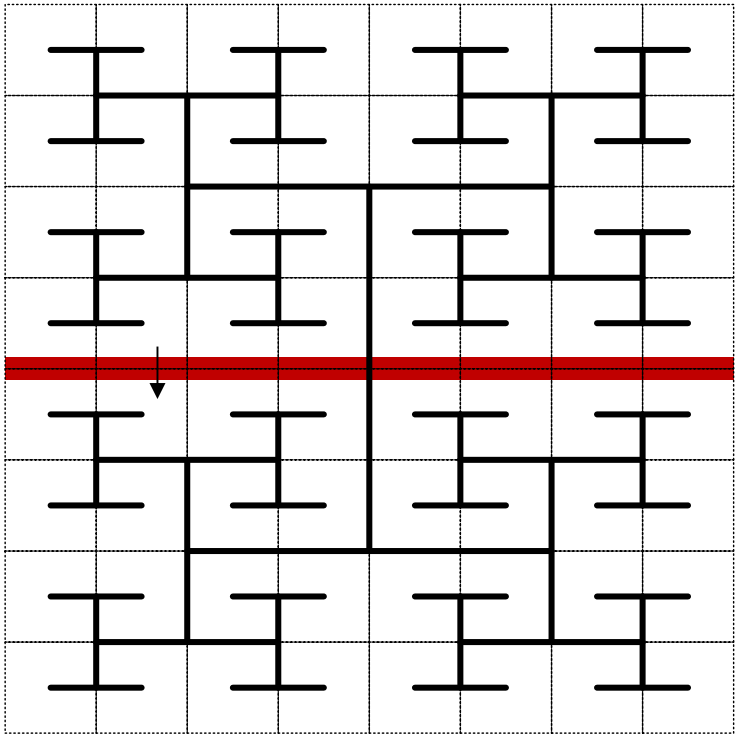
The Performance Implications of Clock Distribution

- All these factors are a function of the path length
  - Reducing the clock tree size reduces clock loss



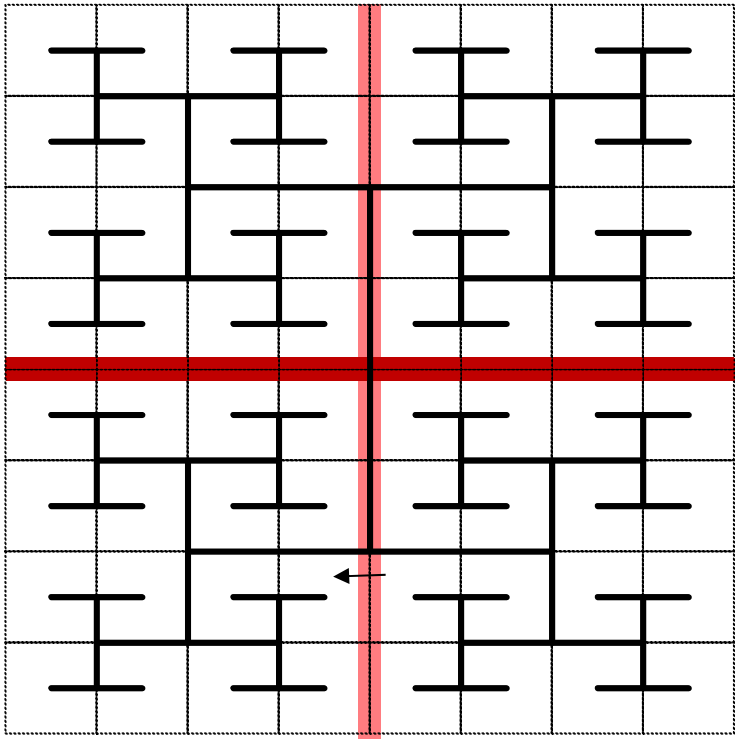
Clock Loss in Fixed Trees

- Clock loss chasms



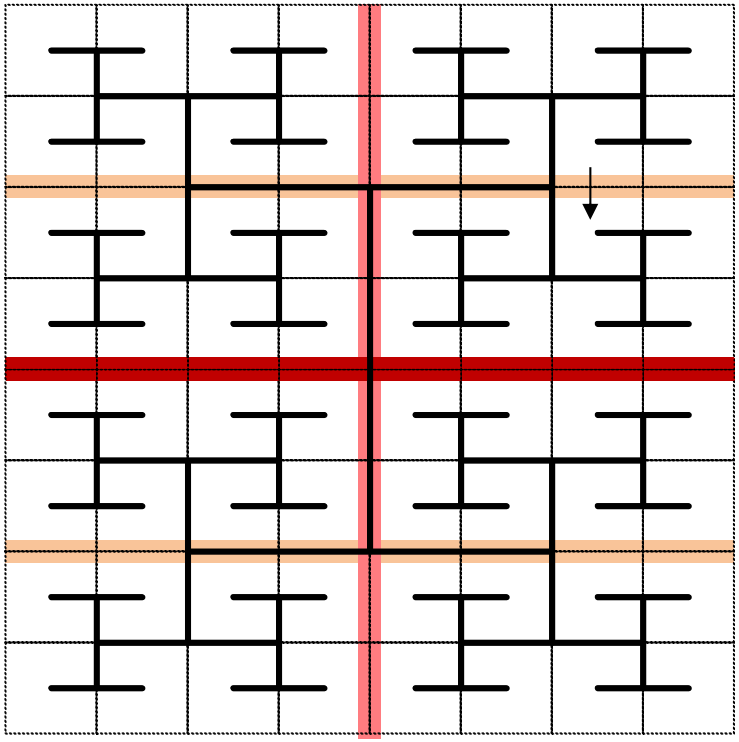
Clock Region Alignment to Fixed Trees

- Clock loss chasms

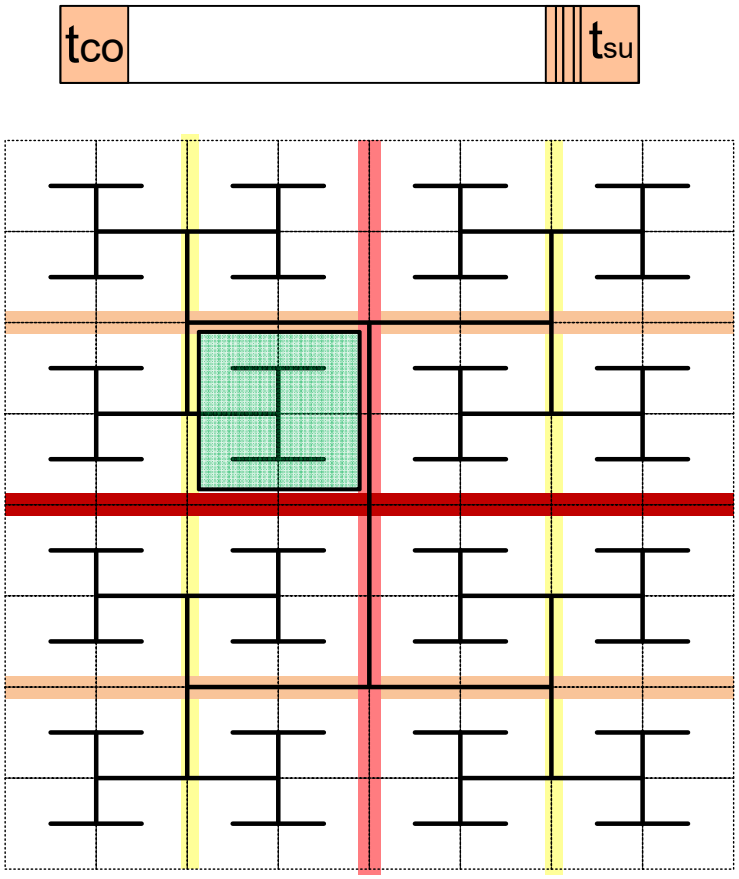


Clock Region Alignment to Fixed Trees

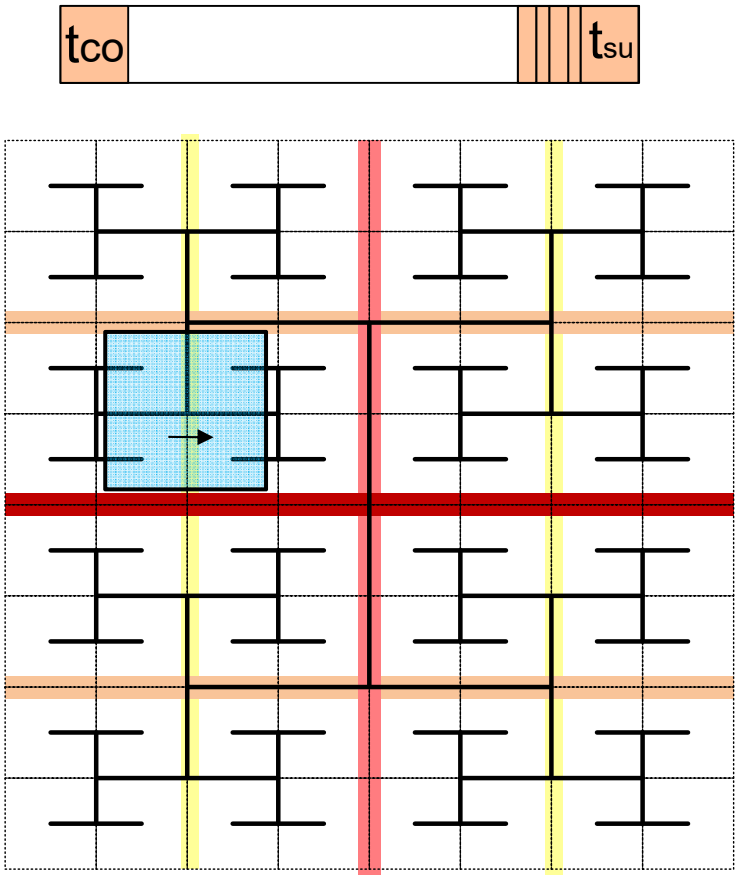
- Clock loss chasms



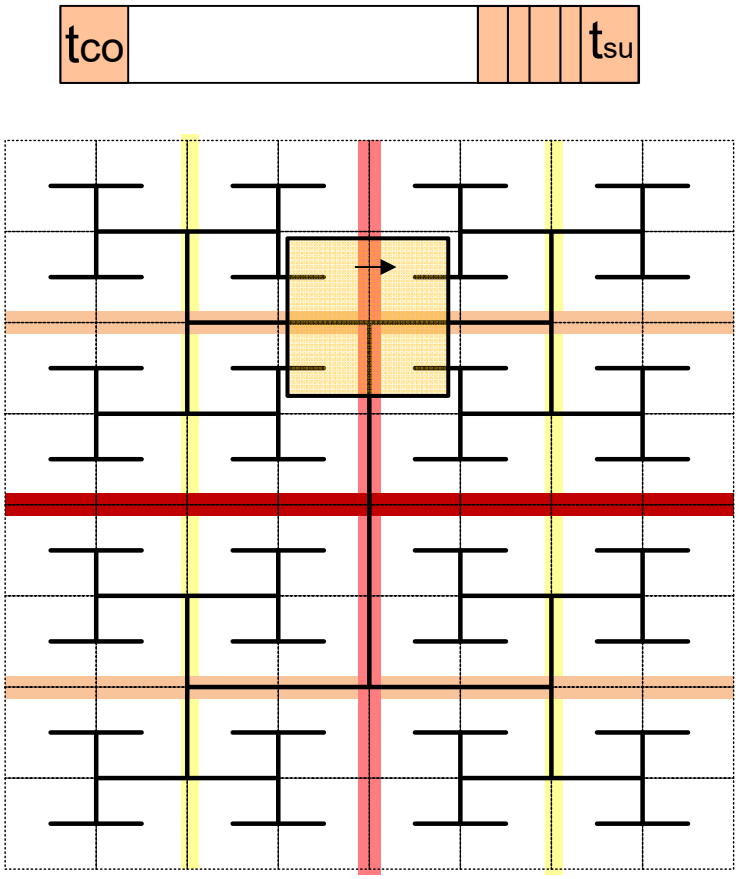
Clock Region Alignment to Fixed Trees



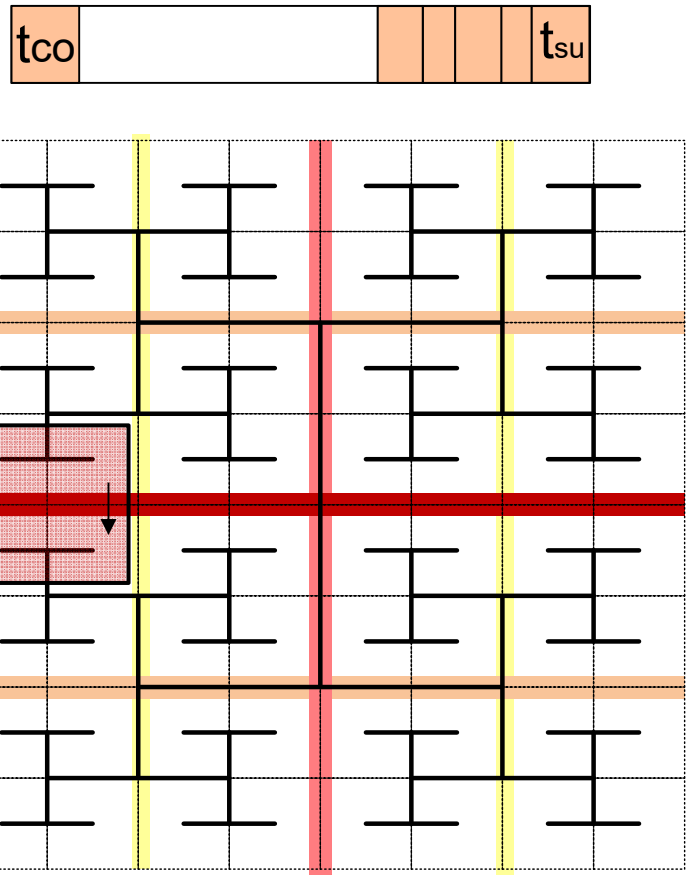
Clock Region Alignment to Fixed Trees



Clock Region Alignment to Fixed Trees

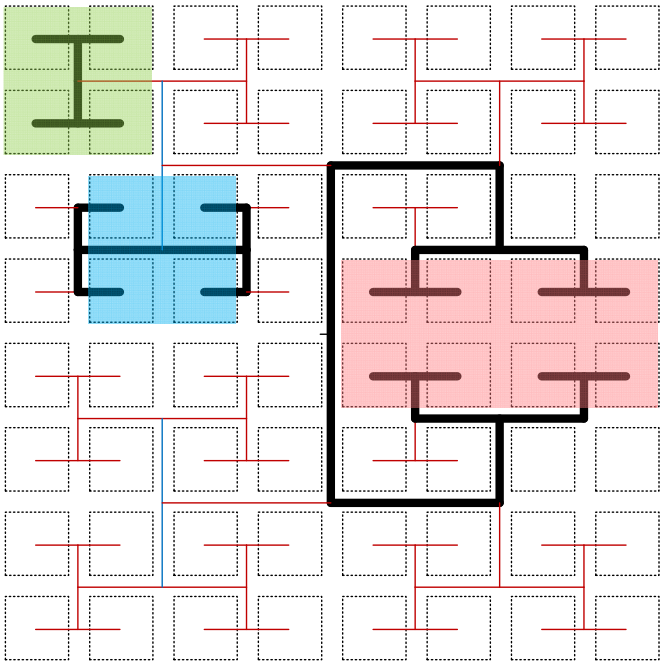


Clock Region Alignment to Fixed Trees

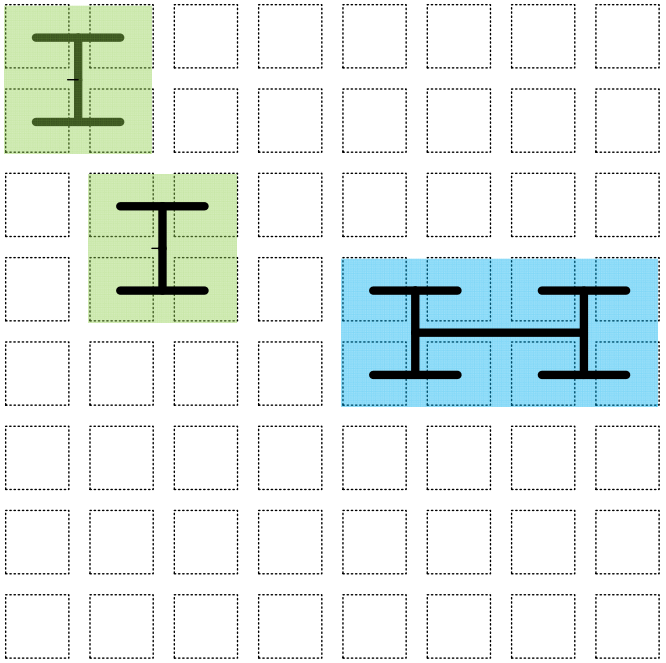


The Advantage of Customized Clock Trees

- Fixed clock trees



Custom clock trees

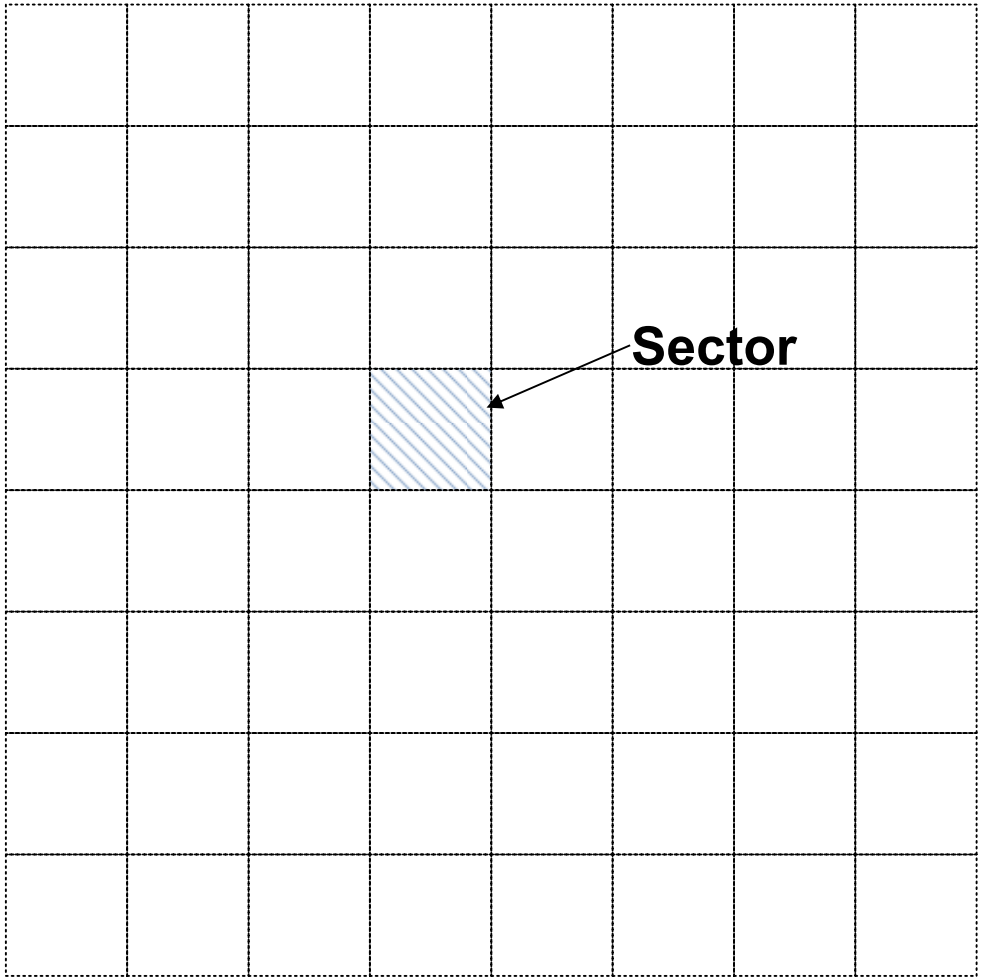


- Several clock regions can share the same clock plane
- Custom clock regions minimize divergent insertion delay and clock loss

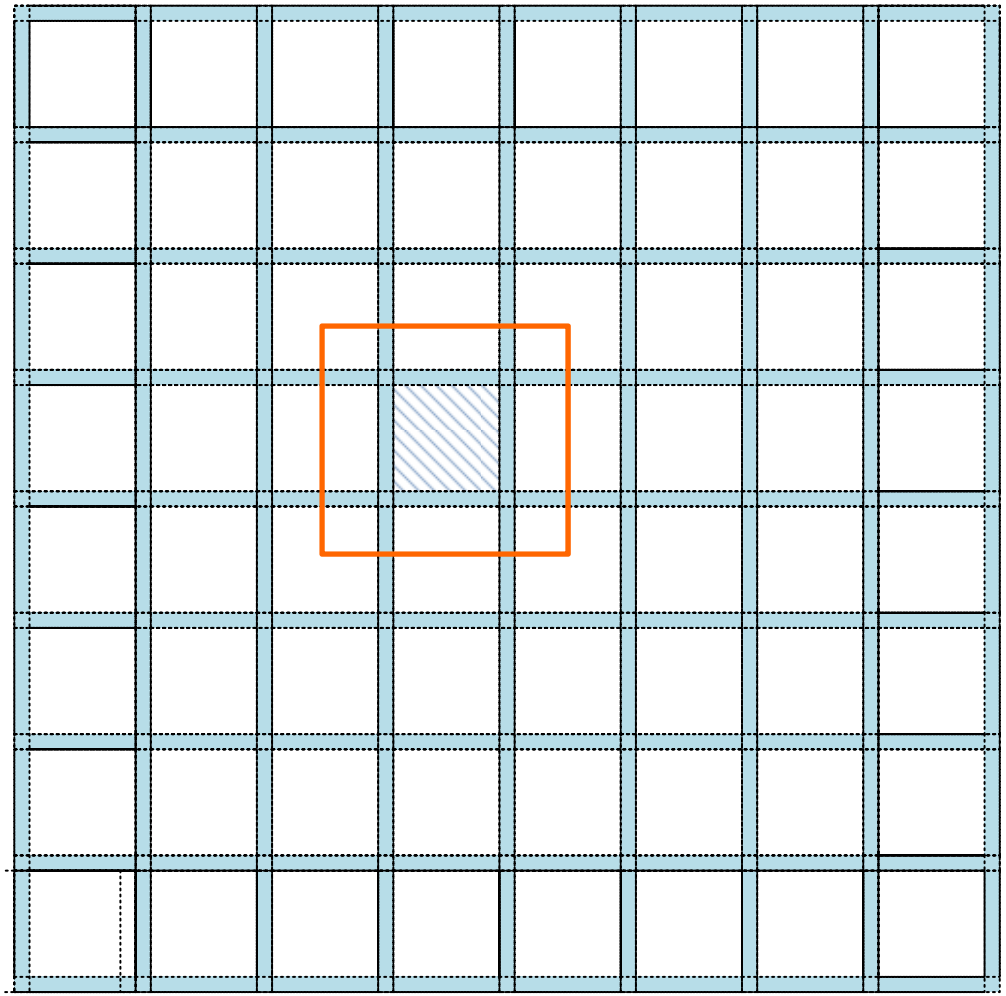
Stratix 10 Routable Clocks

- Construct any clock tree
  - Arbitrary size
  - Arbitrary placement
  - Support different types of tree: H-tree, fishbone, …
- Flexibility has cost and delay implications
  - But in the end, we'll come out ahead

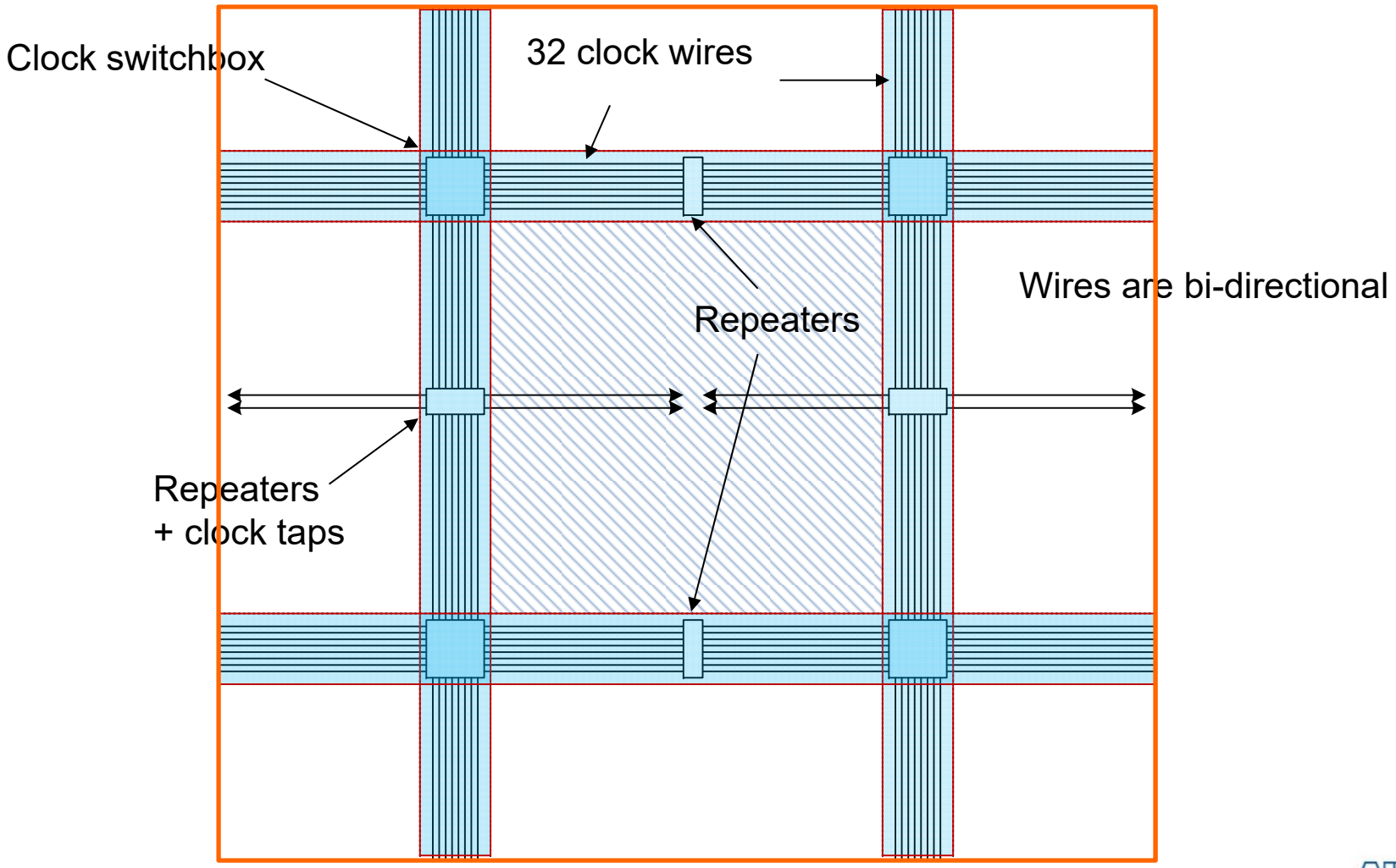
Stratix 10 is an Array of “Sectors”



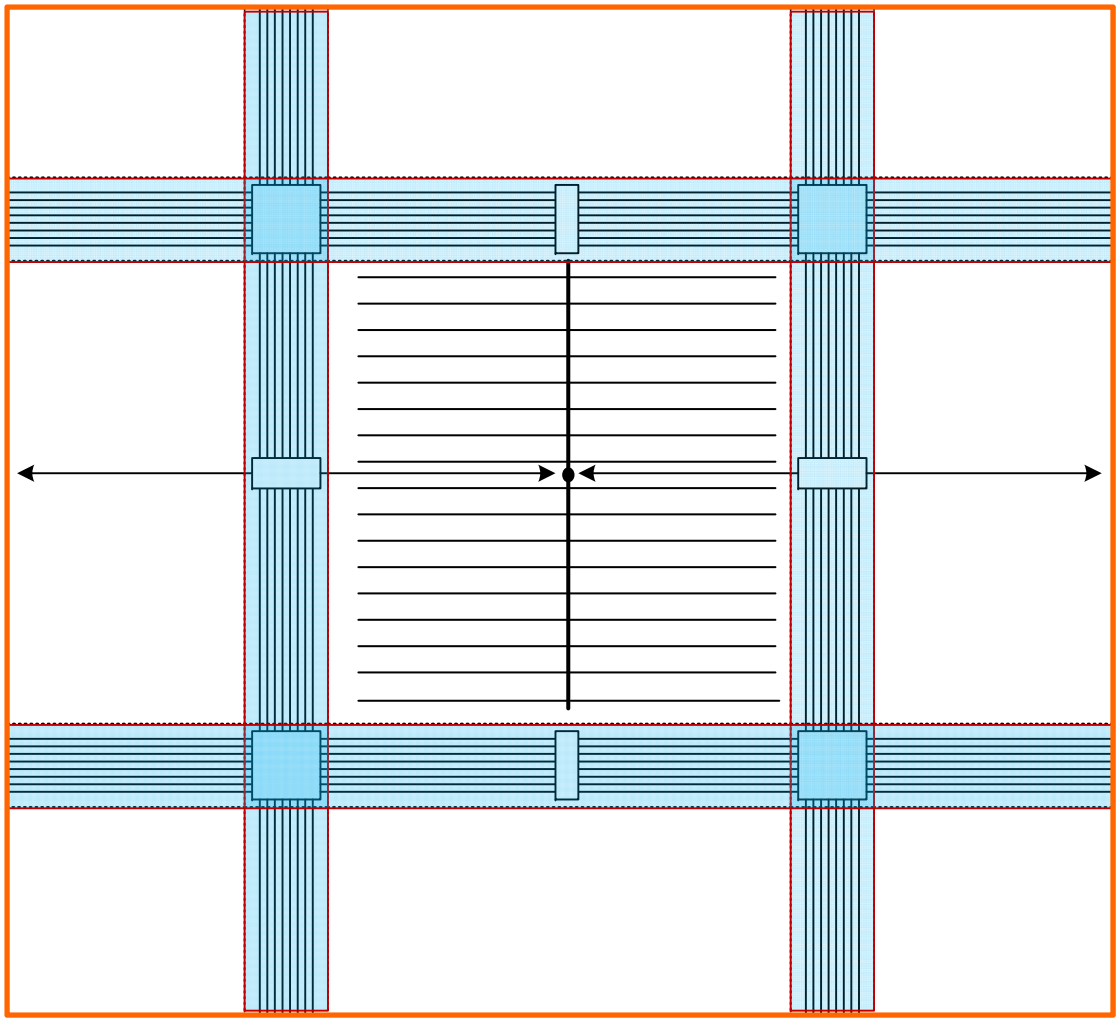
The Routable Clocks are in the Seams between Sectors



Clock Grid



Intra-Sector Clock Distribution

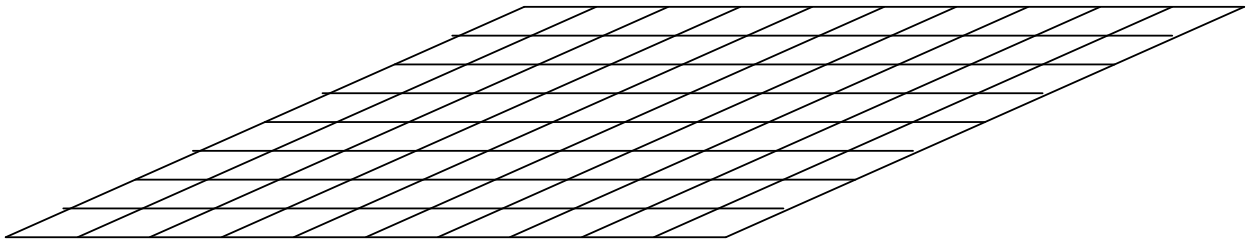


The Routable Clock Challenge

- Provide sufficient flexibility
- With a minimum of added cost and delay

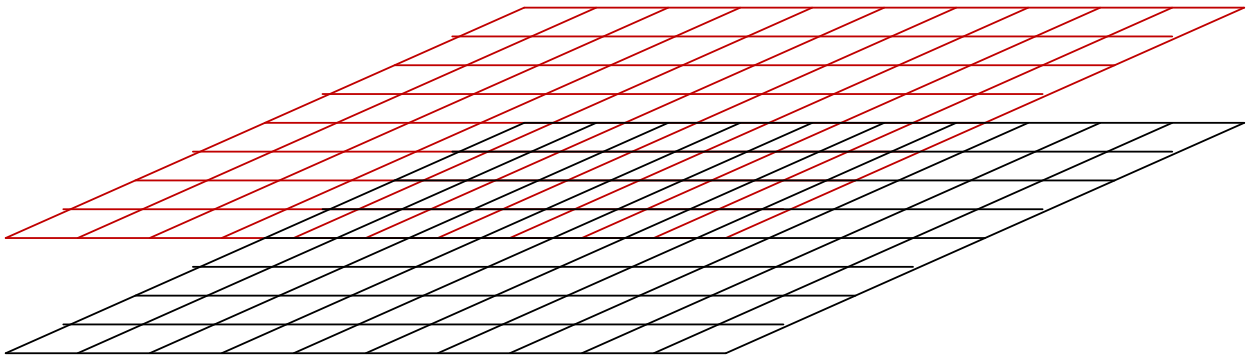
Clock “Planes”

- Label clock segments 0 – 31
  - Each set of segments labeled N forms a plane: 32 clock planes



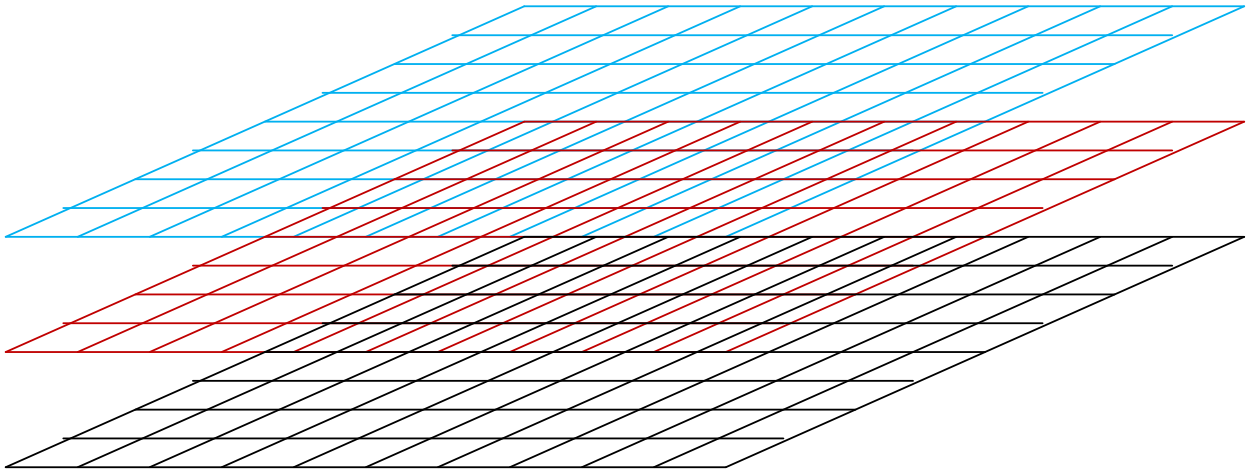
Clock “Planes”

- Label clock segments 0 – 31
  - Each set of segments labeled N forms a plane: 32 clock planes



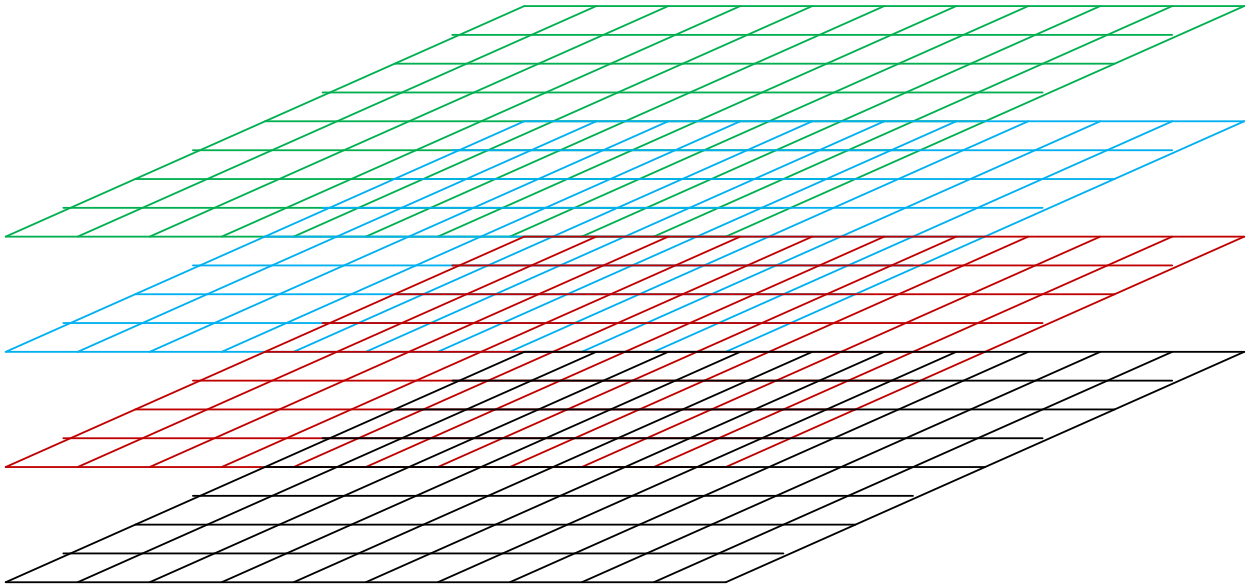
Clock “Planes”

- Label clock segments 0 – 31
  - Each set of segments labeled N forms a plane: 32 clock planes



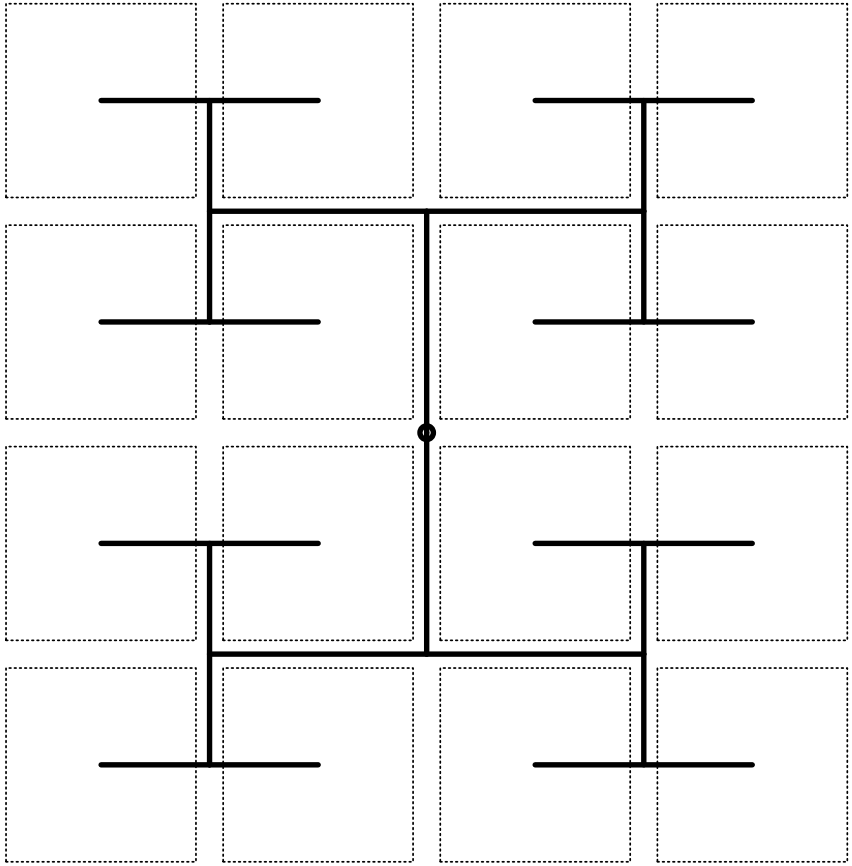
Clock “Planes”

- Label clock segments 0 – 31
  - Each set of segments labeled N forms a plane: 32 clock planes



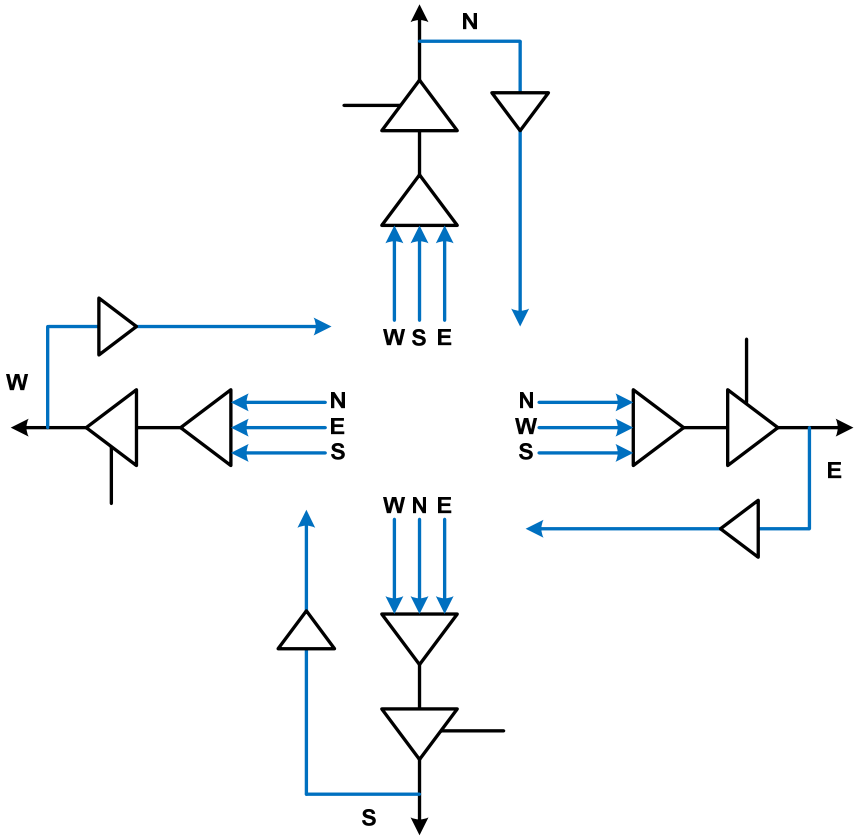
Clock “Planes”

- H-tree can be constructed in one plane
  - No crossovers



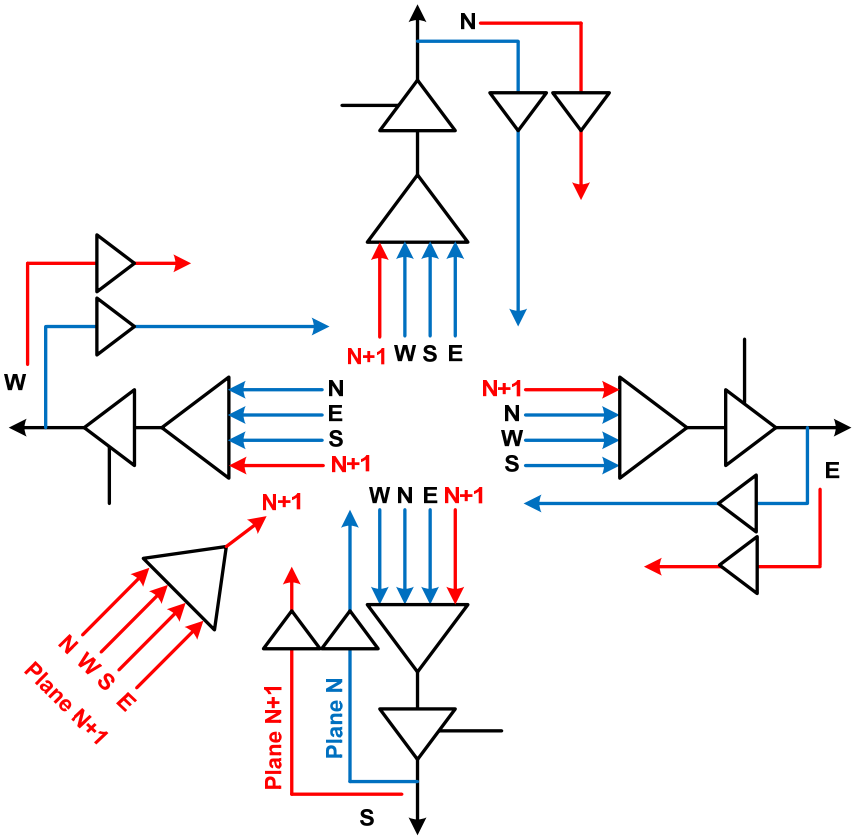
Intra-Plane Muxing

- Simple 3-1 mux on each segment
- Small incremental delay
- Small cost



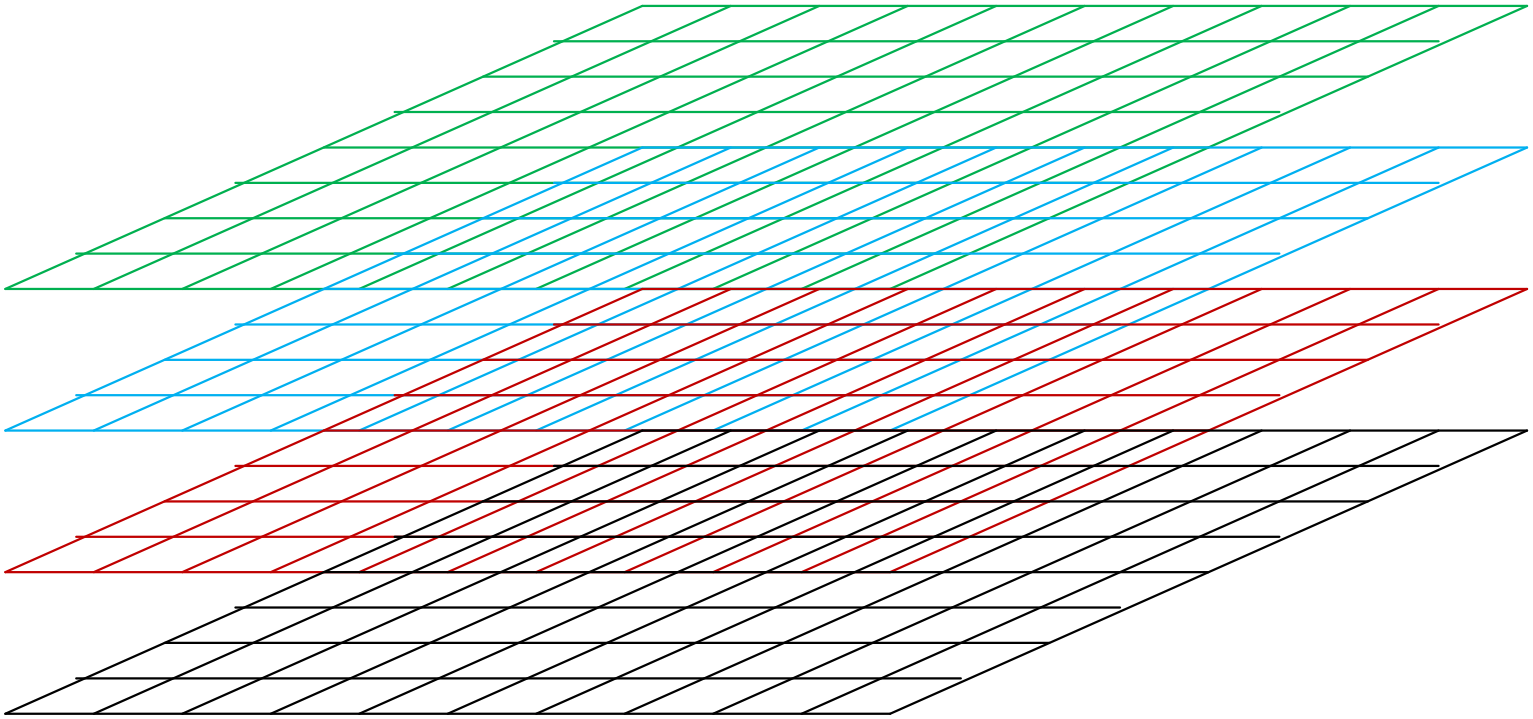
Inter-Plane Muxing

- Must support cross-overs
  - Route from source to tree root
  - More complex clock trees
- Add one more multiplexer
  - Connect plane N+1 to plane N



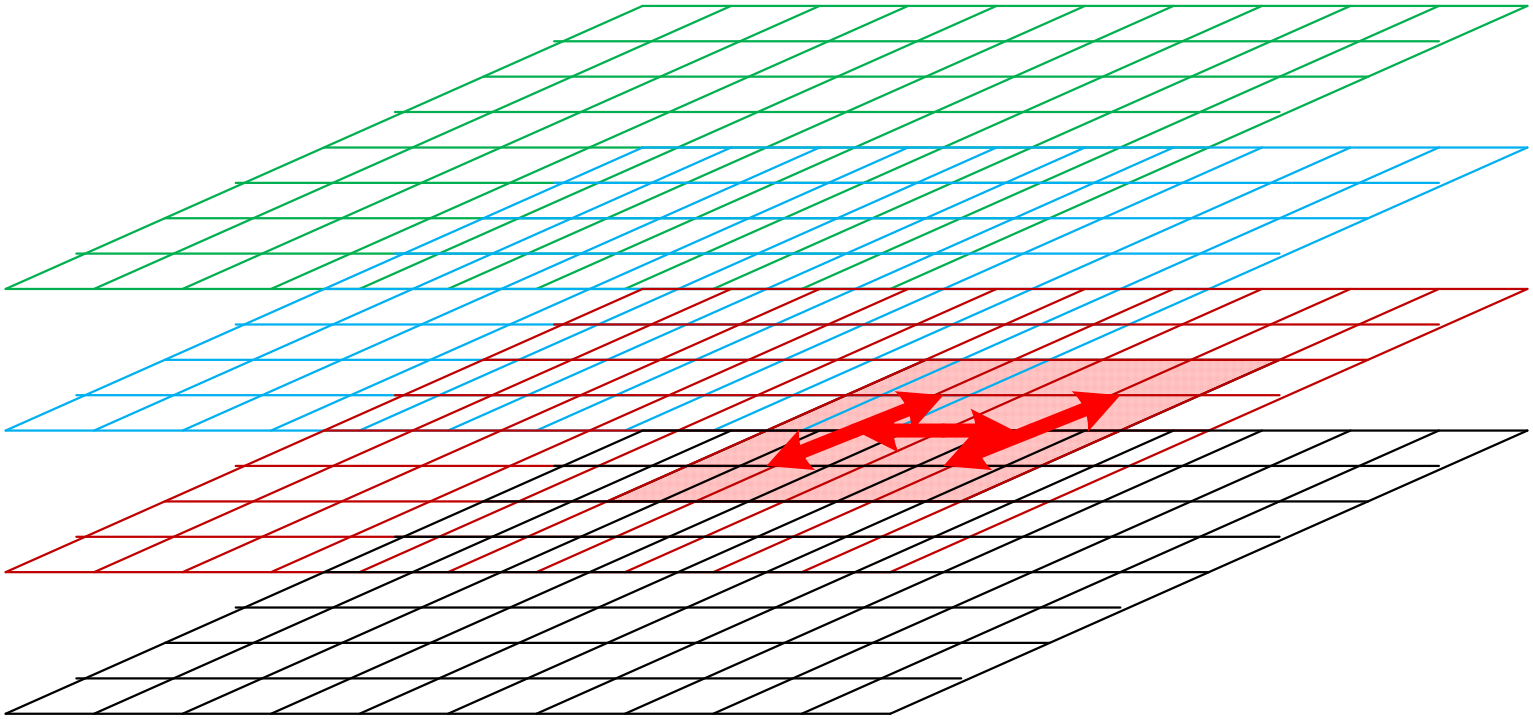
Clock Tree Synthesis

- Identify the clock region



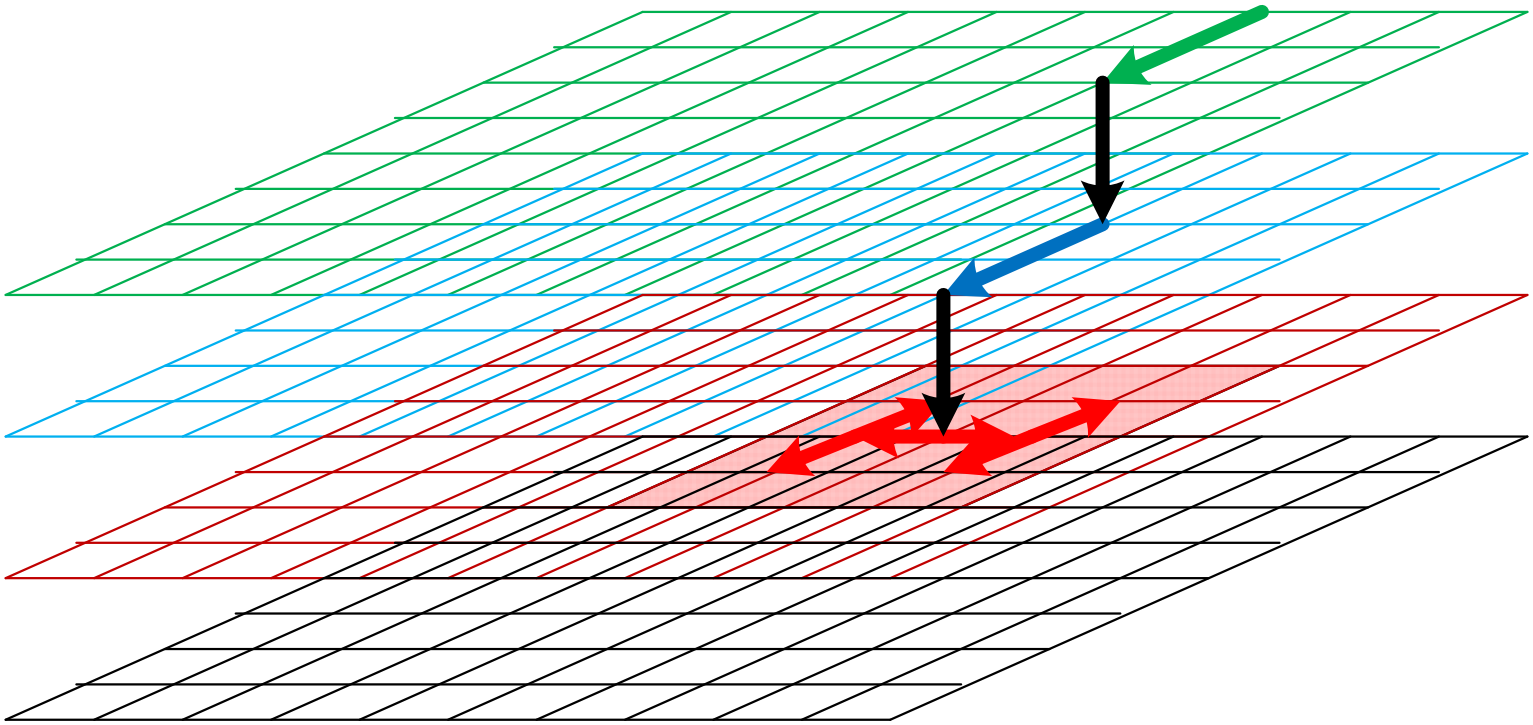
Clock Tree Synthesis

- Identify the clock region
- Choose a clock plane and generate a balanced clock tree



Clock Tree Synthesis

- Identify the clock region
- Choose a clock plane and generate a balanced clock tree
- Route clock from source to root of clock tree

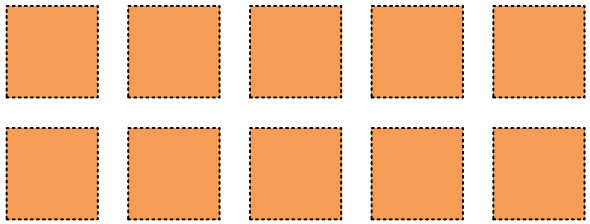


Non-canonical Clock Trees

- Clock regions are generally not square
  - Nor 2^n x 2^m
- What about trees of arbitrary size and shape?

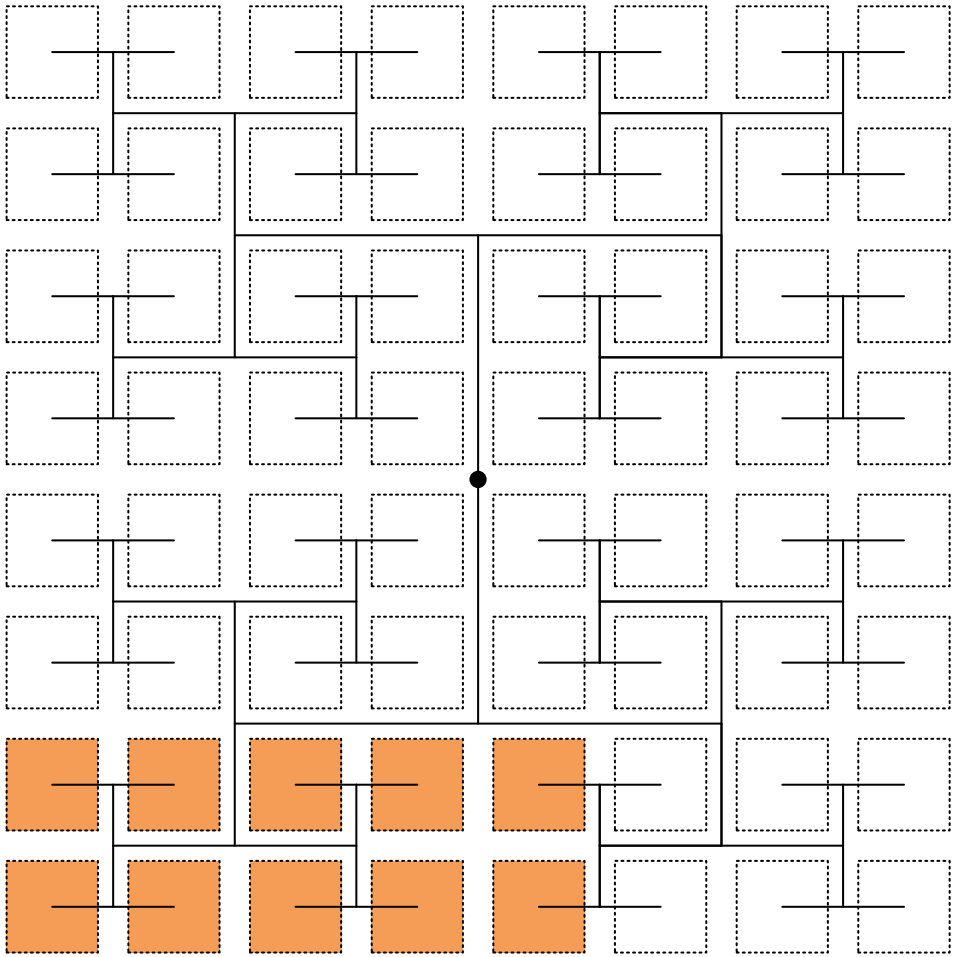
Clock Tree Synthesis Algorithm

- Generate minimal height tree
- Example: 2 x 5 clock region



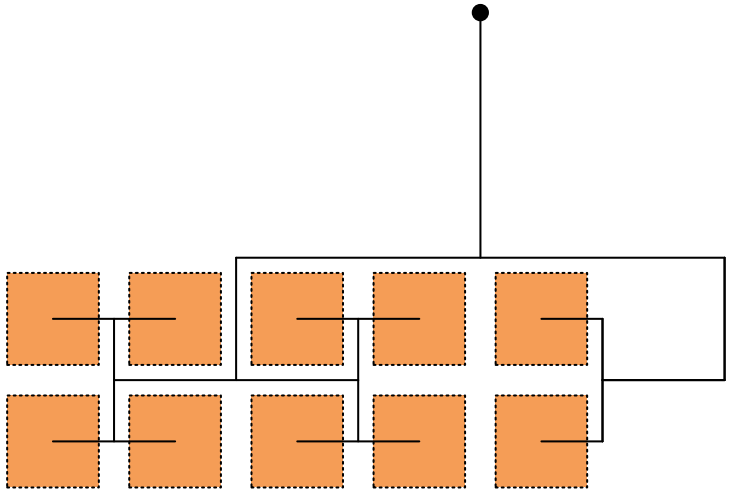
Clock Tree Synthesis – Graphical Representation

- 1. Cover with smallest enclosing 2^n x 2^n tree



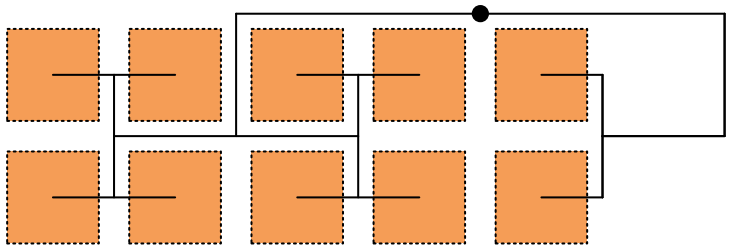
Clock Tree Synthesis – Graphical Representation

- 2. Prune tree

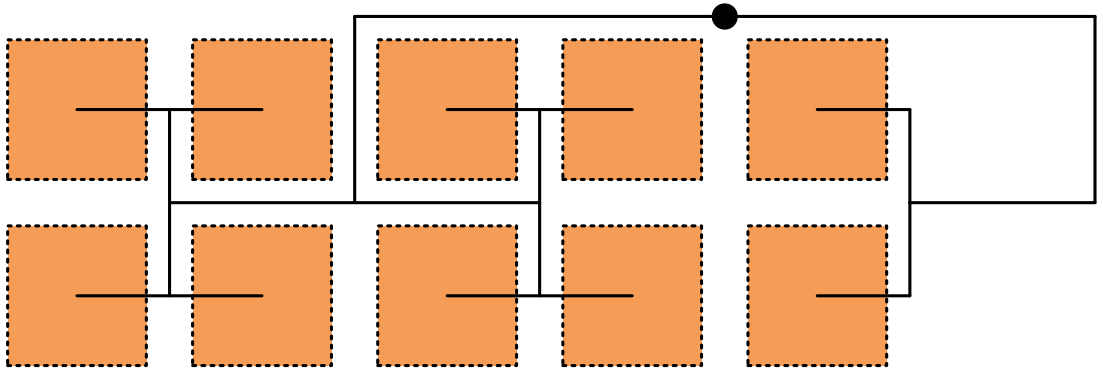


Clock Tree Synthesis – Graphical Representation

- 2. Prune tree

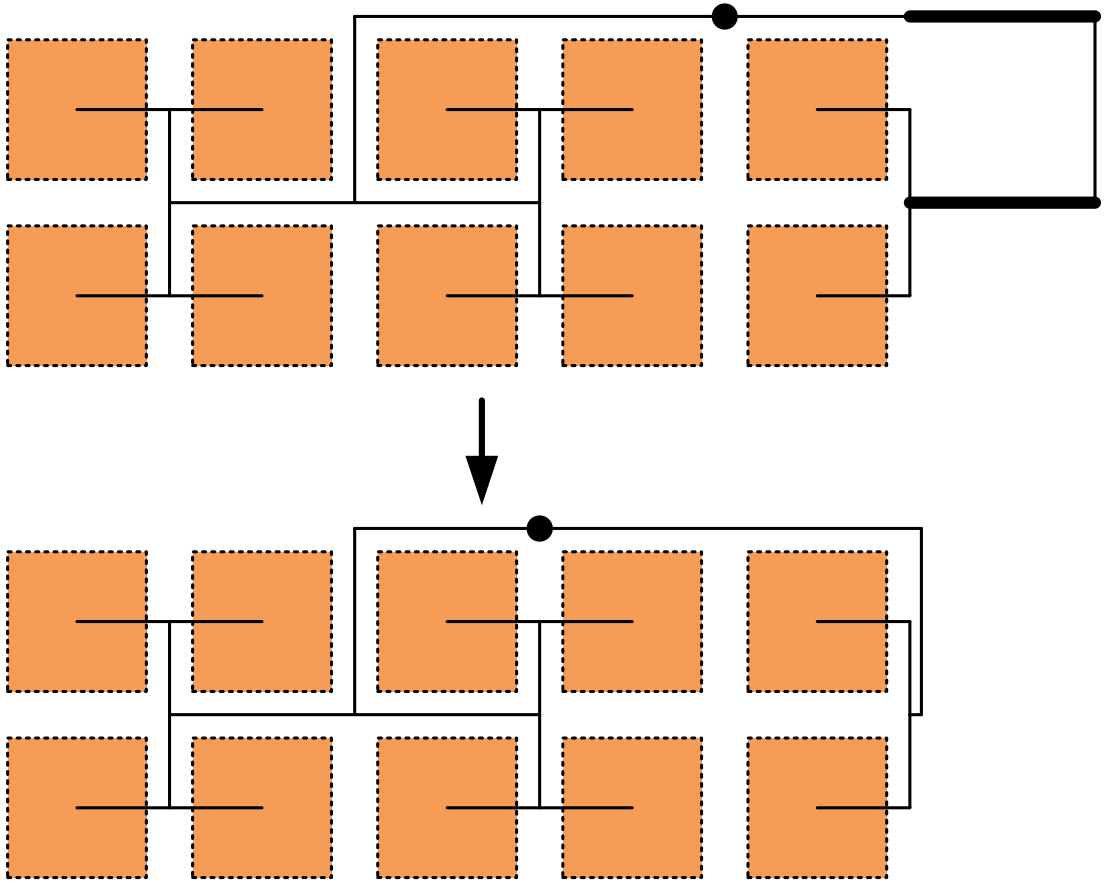


Clock Tree Synthesis – Graphical Representation



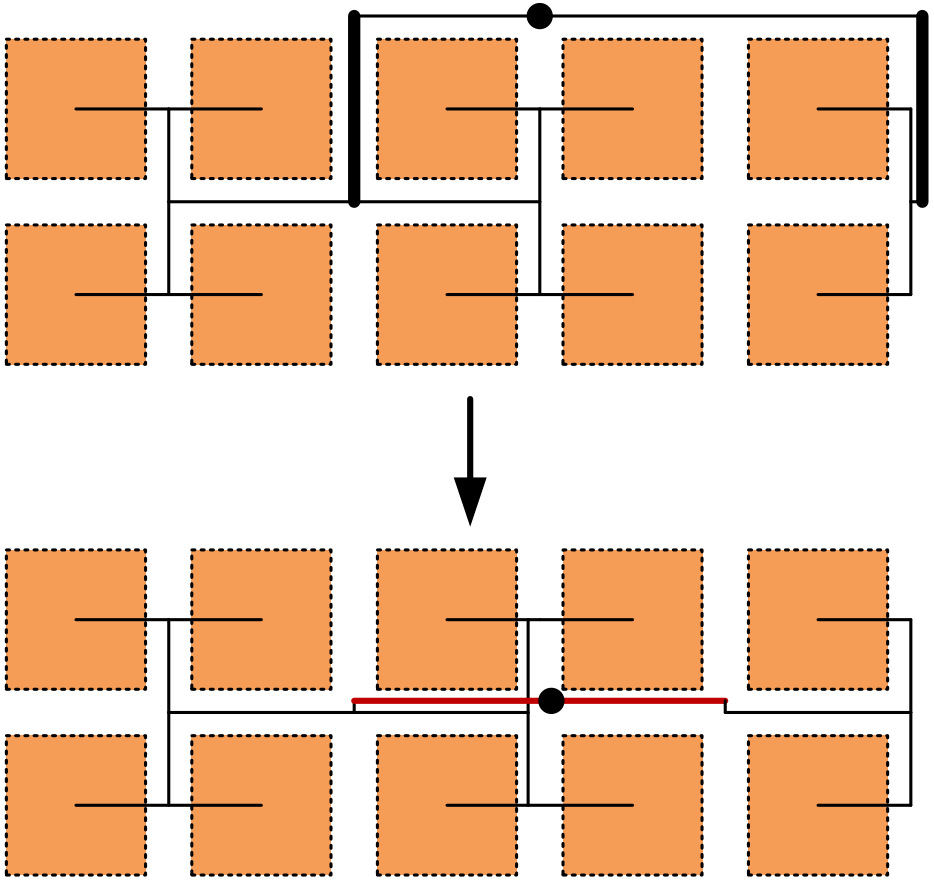
Clock Tree Synthesis – Graphical Representation

- 3. Remove redundant segments in subtree



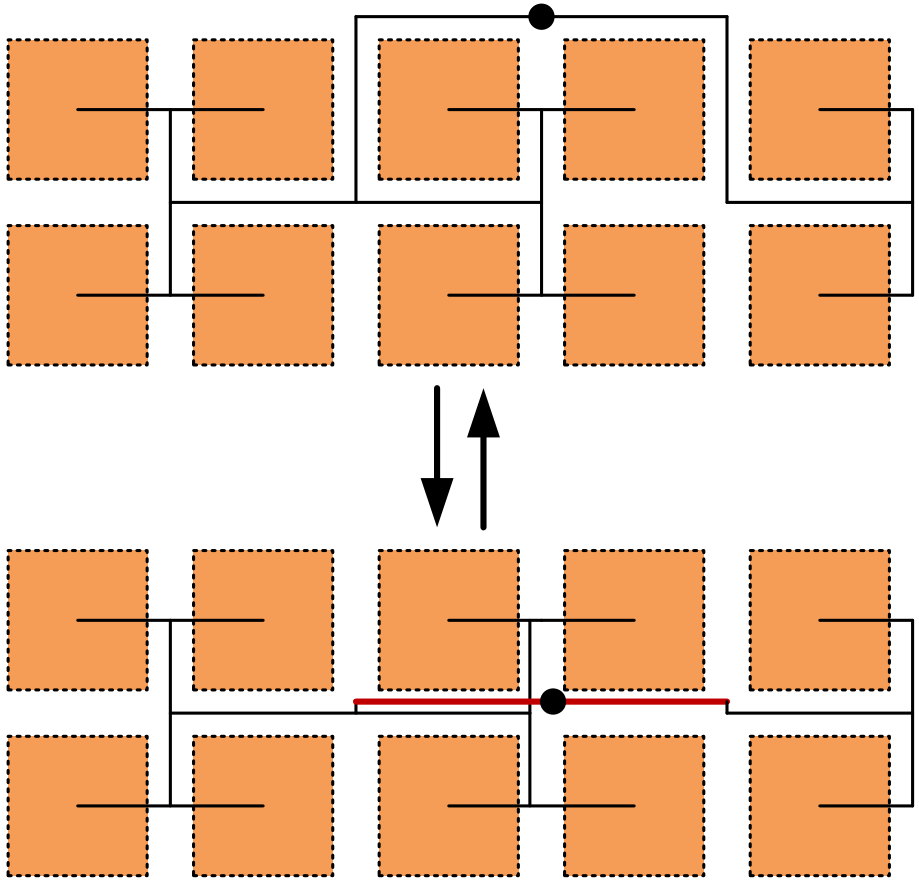
Clock Tree Synthesis – Graphical Representation

- 4. Reduce tree height



Clock Tree Synthesis – Graphical Representation

- Tradeoff tree height for less overlap



Clock Tree Delay

- Most connections go through a single primary mux
  - Minimal additional delay
- A few go through additional secondary mux
  - Where overlap occurs in the tree
- Path delays are inherently balanced
  - Leaves are all on plane N, at the same depth
  - All paths from source (plane N+k) to leaves go through the same number of primary and same number of secondary muxes

Clock Insertion Delay

- Flexibility does add insertion delay
- Bi-directional clock wires add some delay
  - Tradeoff delay for cost
- Total delay relative to fixed clocks:
  ~1.3-1.4x delay for same distance
- What is the effect on performance?
  - Routable clock tree can be much smaller than fixed trees
  - But for the same size, fixed trees have less clock loss

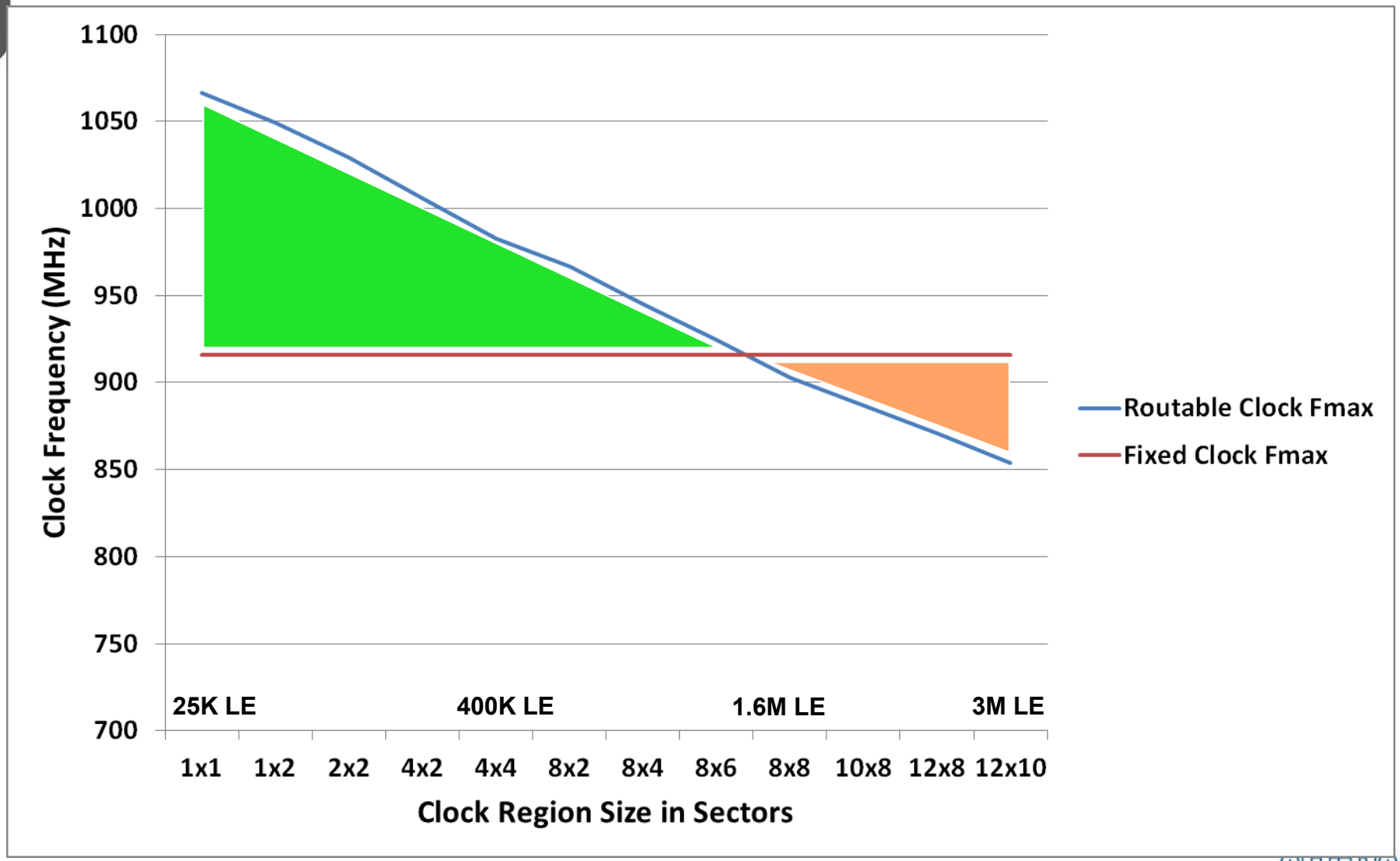
Performance Analysis 1

- Clock loss modeled as a linear function of insertion delay
- Compare performance based on clock tree size
- Assumptions:
  - Some critical path must cross the worst-case skew chasm

- Example:
  Critical path = 760ps ： What Fmax can I expect?

  Depends on the size of the clock region

Performance vs. Clock Region Size (760ps critical path)

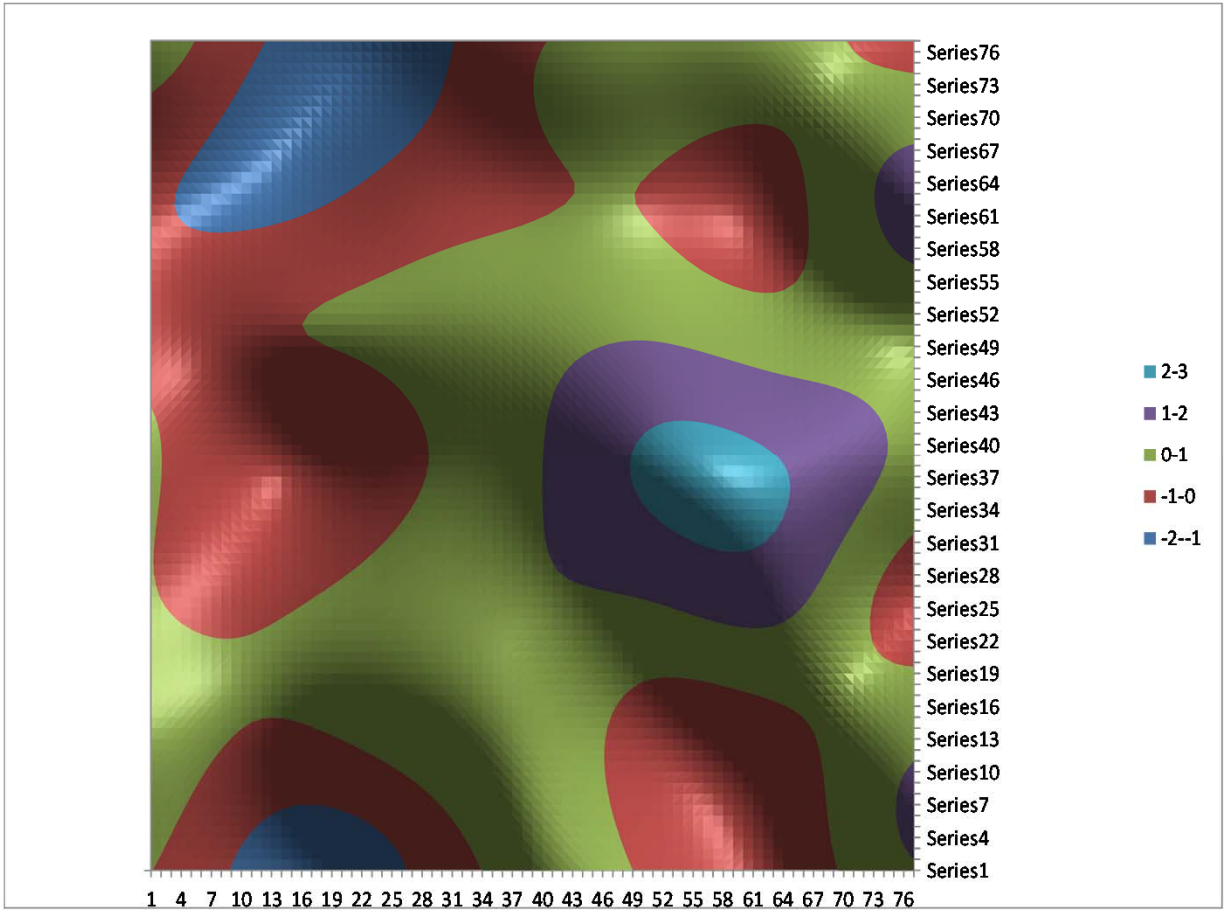


Performance Analysis 2

- Monte Carlo analysis
- Large set of benchmarks
  - From previous generation FPGA
- Generate a variation profile according to model
- Locate a design randomly on the die
- Compute Fmax
  - 1. Using a fixed clock tree
    2. Using a custom clock tree for clock region
- Repeat for many designs, variation profiles and placements

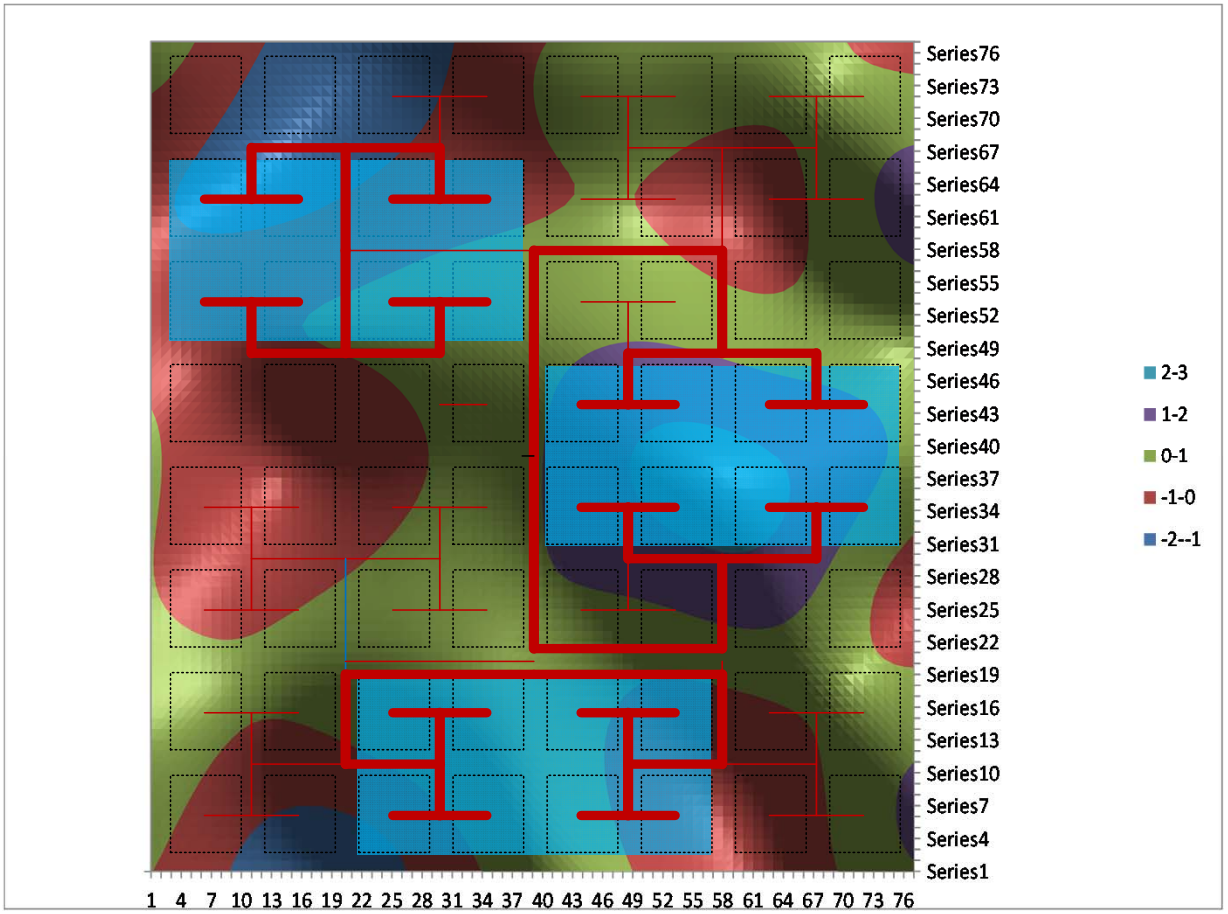
Example Variation

- Analysis captures effect of variation correlation



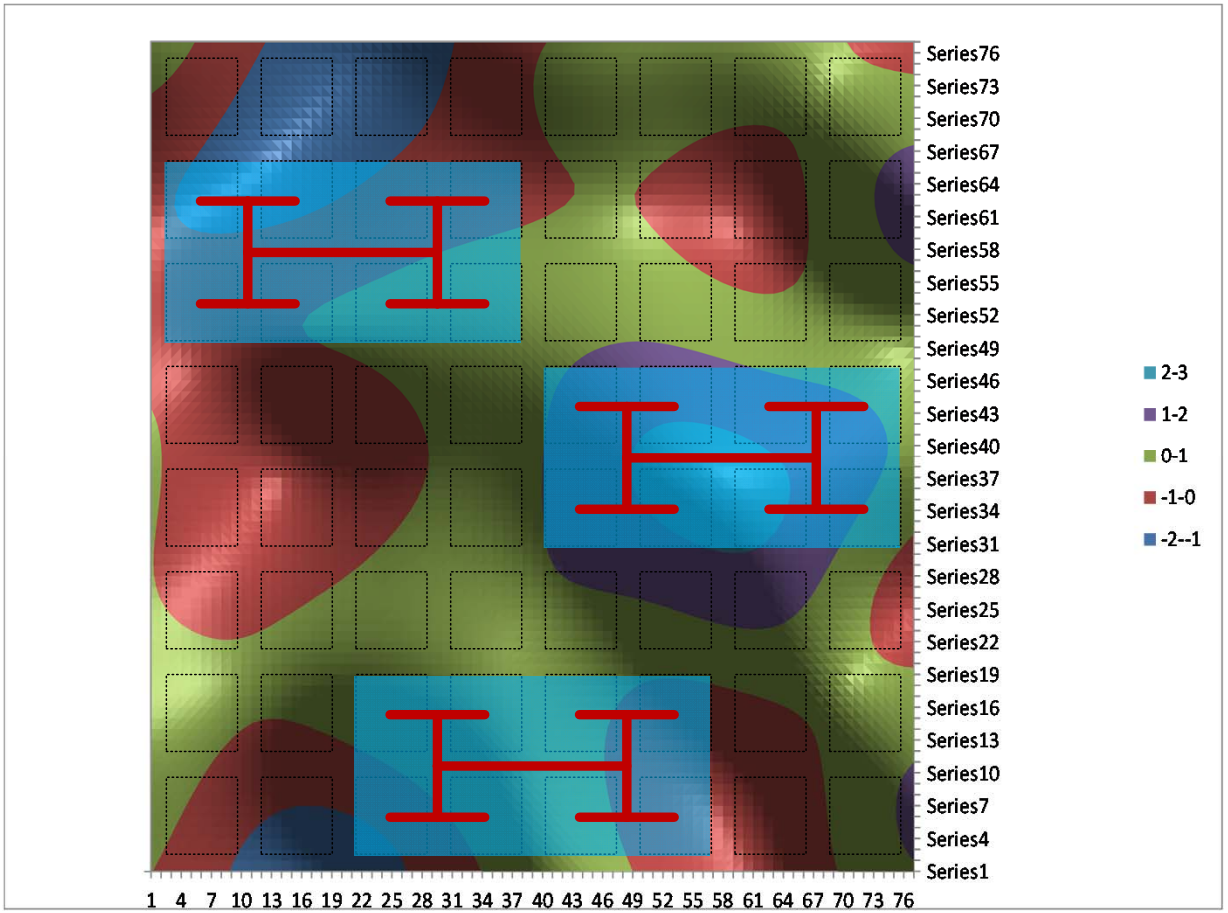
Monte Carlo Experiment

- Map design to random locations with fixed Clocks

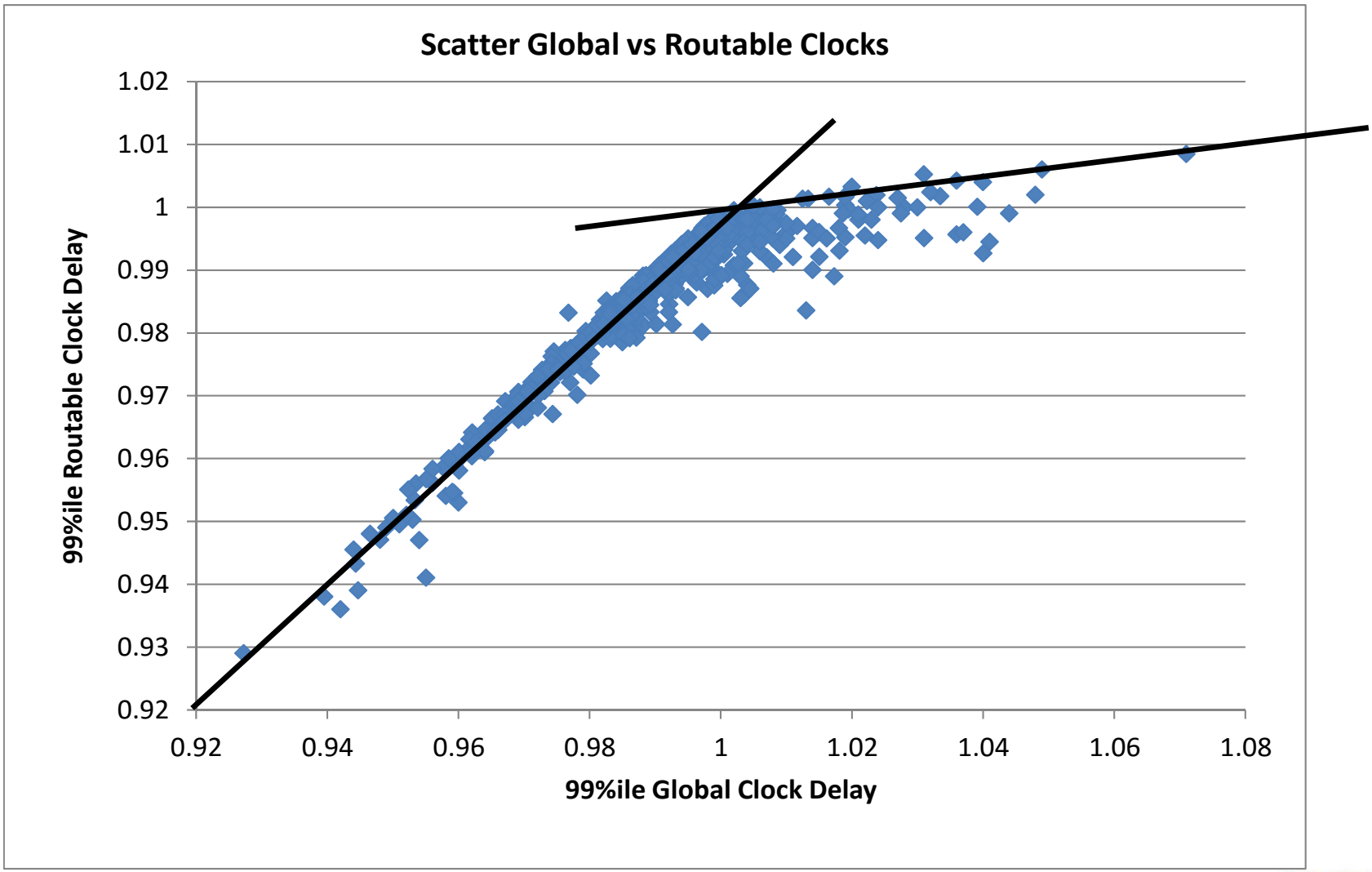


Monte Carlo Experiment

- Map using routable clocks



Results – Routable Clocks Reduce Worst Case by 6%



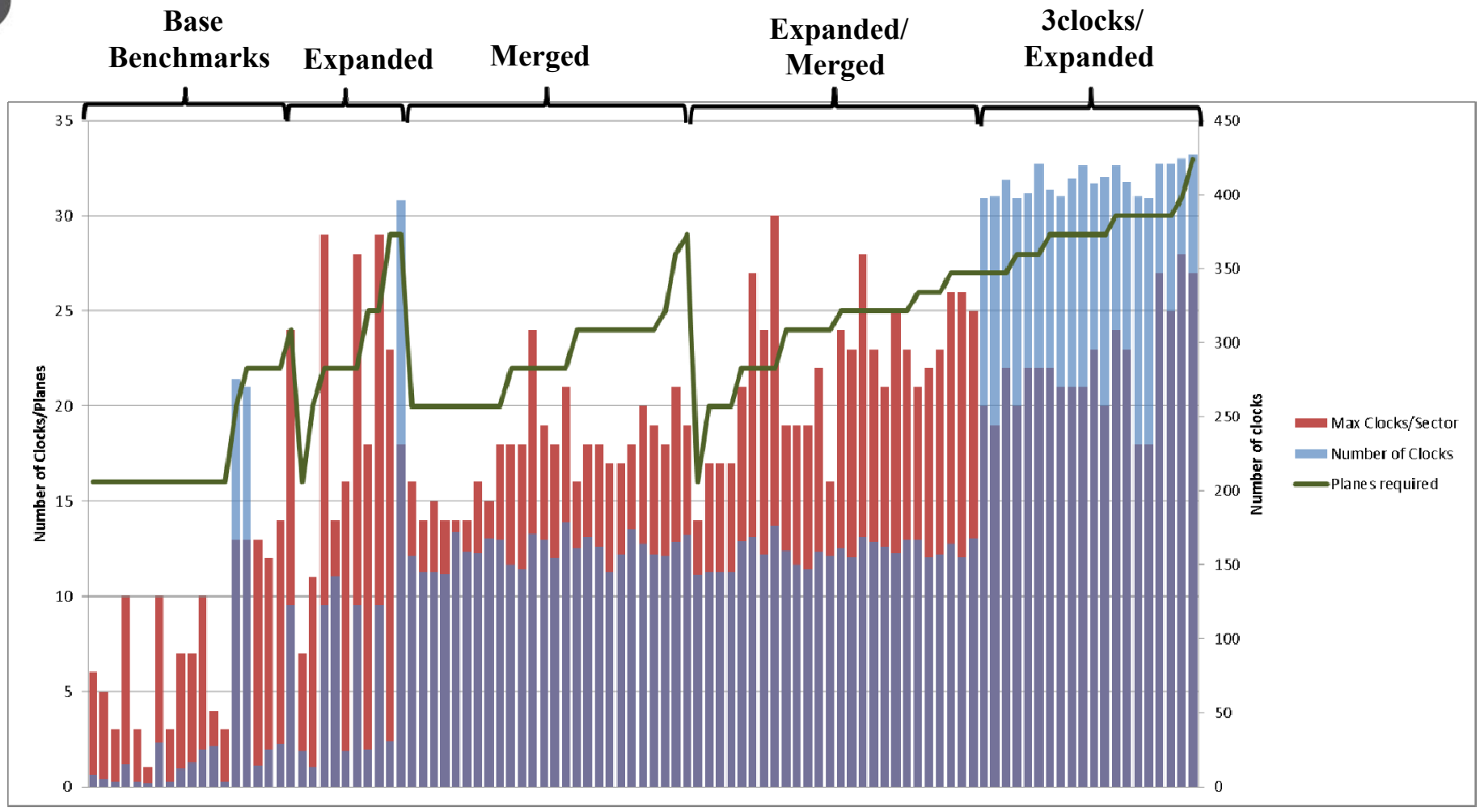
Capacity

- Does the clock grid with 32 planes provide enough clocks?
- Some designs use well over 100 clocks

Capacity Experiments

- Used large Arria 10 designs with large clock demands
- Extended the benchmarks
  - scaled to Stratix 10
  - merged designs
  - added many local transceiver clock regions
- Mapped each clock region on Arria 10 to corresponding set of sectors on Stratix 10
- Used prototype clock tree synthesis tools
  - Optimal, balanced clock trees
- Generated clock trees using N clock planes
  - N = 16 – 32

Capacity Results



Stratix 10 Clock Network Summary

- Transition to custom-built clock trees
  - Scalability
  - Performance
- Challenges:
  - Provide sufficient flexibility
  - Without too much delay overhead
- Performance difference averages about 6%
  - But can be larger if designs are partitioned well

Future Work

- ASIC design methodologies for FPGAs
- Many opportunities for CAD tools
  - Design partitioning
  - Clock partitioning
  - Iterative floorplanning, placement, clock tree synthesis
  - Skew-aware place and route
  - Take advantage of “Interesting” clock tree structures

Future Work

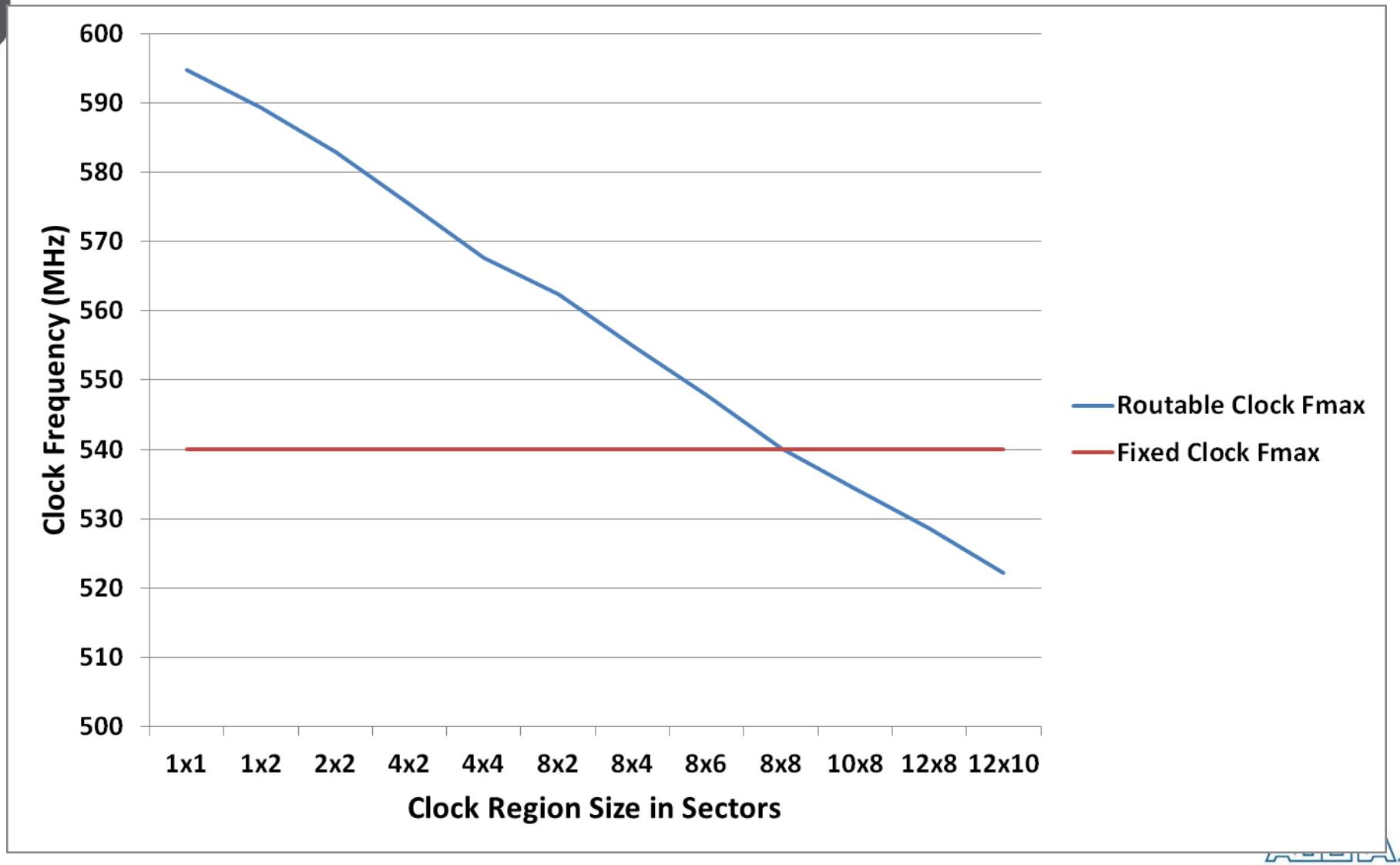
- ASIC design methodologies for FPGAs
- Many opportunities for CAD tools
  - Design partitioning
  - Iterative floorplanning, placement, clock tree synthesis
  - Skew-aware place and route
  - Take advantage of “Interesting” clock tree structures
- Buy Stratix 10!

Thank You

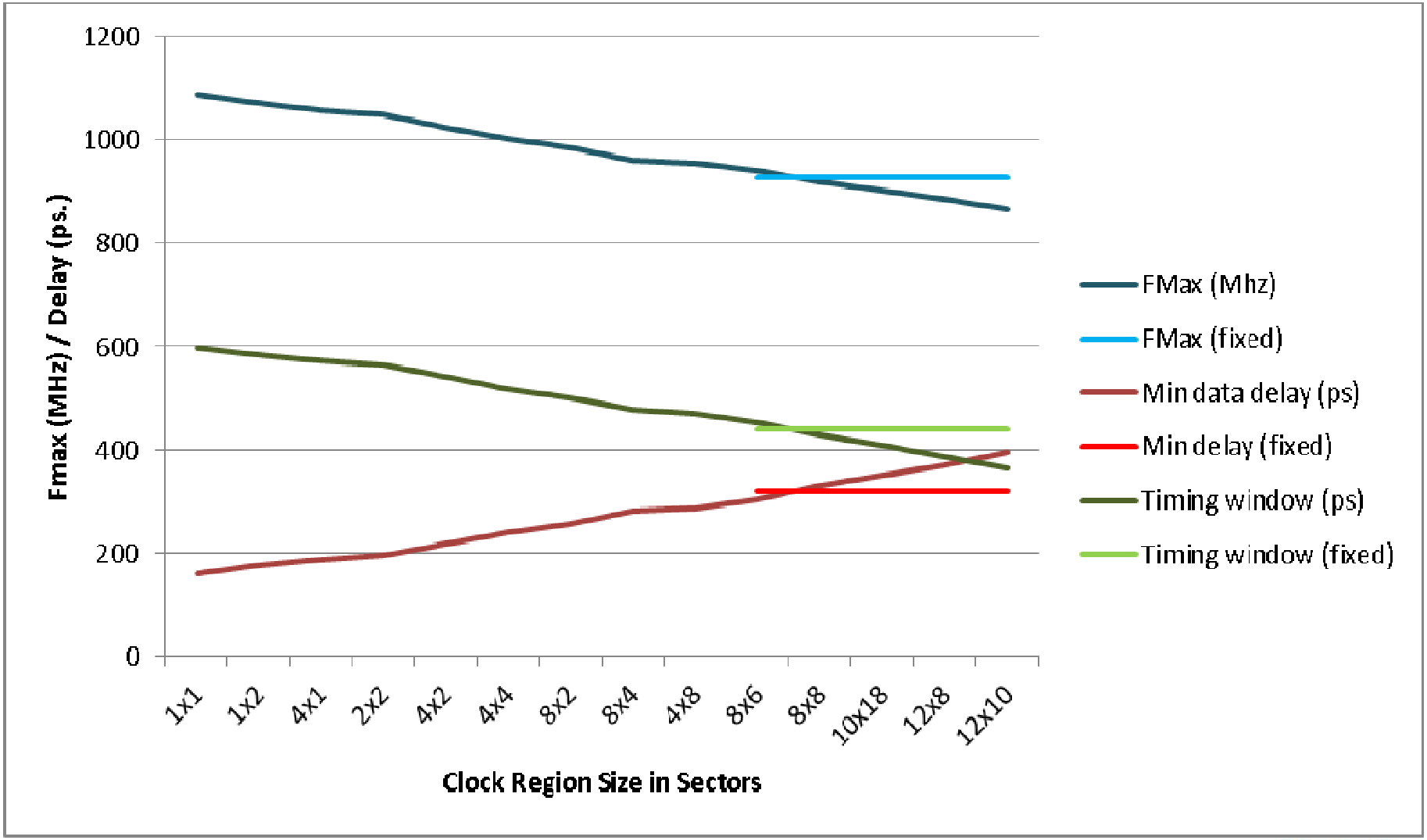
Performance Comparison – Fixed vs. Routable Clocks

- Infeasible to run real experiments
  - No benchmarks, no CAD tools
- Model performance based on size of clock region
  - Fixed global clocks vs. customized clocks
- Change in design methodology
  - Global clocks kill performance
  - Long recognized in ASICs
- Partition design into communicating clock regions
  - GALS: Globally Asynchronous, Locally Synchronous
  - May be meso-synchronous (same clock, arbitrary skew)
- Sectors are large! ~25K LEs

Performance vs. Clock Region Size (1200ps critical path)



Performance vs. Clock Region Size (800ps critical path)



Related Work

- Very little academic work
  - Limited to fixed clocks with routable source-to-root routing
- Xilinx Ultrascale clocks
  - “Clock region” tiles
    - Similar to sectors, but are not uniform
  - Two separate clock networks:
    - “Routing”: Route source to clock tree
    - “Distribution”: Build local clock tree
  - Clock tracks run through the center of CRs
  - Trees not inherently balanced
  - Seems biased towards fishbone clocks: Rib-and-Spine
  - Includes tunable delays