Tier Partitioning and Flip-flop Relocation Methods for Clock Trees in Monolithic 3D ICs

Da Eun Shim[1], Sai Pentapati[1], Jeehyun Lee[1], Yun Seop Yu[2], and Sung Kyu Lim[1]
[1]School of Electrical and Computer Engineering, Georgia Institute of Technology, USA
[2]Department of Electrical, Electronic, and Control Engineering, Hankyong National University, Korea

***Abstract*—In this paper, we propose simple but effective clock tree optimization algorithms for monolithic 3D ICs that are based on tier partitioning and flip-flop relocation. Our algorithms take into account 3D timing critical paths, clock skew, and the clock tree hierarchy for a better quality 3D clock tree. We also perform clock slew manipulation and buffer reduction to further improve the 3D designs. We tested four industrial benchmarks implemented using a commercial library and observed up to 34.3% clock skew, 35.9% clock wirelength, 10.0% combinational clock power, and 15.5% total power savings compared to the state-of-the-art [8].**

I. Introduction

With 2D integrated circuits (ICs) gradually reaching their limits due to device scaling challenges, the advent of 3D ICs shows potential to be the new driving force to continue Moore's law. With shorter physical connections between cells, 3D ICs stack dies with smaller footprints on top of each other. With monolithic inter-layer vias (MIVs), the connections between the dies are cheaper and a more fine-grained integration can be implemented. With developments in machine learning/artificial intelligence (AI) and virtual reality (VR), power reduction and performance boost are becoming more important in IC design.

There have been various efforts throughout the years to improve the quality of clock delivery networks. The clock distribution network/clock tree affects both the power consumption and the performance of the IC. As the clock tree delivers clock signals to all the flip-flops in a chip, any delay or skew can cause delays in the path and thus affect the critical path of the chip. The switching and internal power consumed by the clock tree is also a non-trivial amount. Thus, overall a better quality clock tree leads to better full chip quality.

In 2016, Wu et al. used the weighted k-means algorithm to cluster and move flip-flops closer to the cluster centers in 2D ICs [3]. Their algorithm also balances the size of each cluster to be within a certain range. This has proven to be useful for improving clock tree quality. Also in 2016, Kahng et al improved the flop-tray based design which has also shown promise in power reduction [5].

With 3D ICs showing great potential for further improvement, efforts to improve the 3D clock tree have also been seen. In 2012, Zhao developed a new power- and slew-aware 3D clock router that does not make use of the commercial 2D EDA tools [9]. In 2014, along with the development of the state-of-the-art 3D IC shrunk2D flow, Panth et al. fixed the whole clock tree backbone on one of the tiers while partitioning the leaf cells in the clock tree (flip-flops) into different tiers using area balance [7]. This was one of the first efforts to use commercial 2D routers to improve 3D clock tree quality.

As an effort to further improve power savings while using a commercial tool, we propose a novel clock tree clustering and flip-flop relocation method for M3D that makes use of 2D commercial EDA tools. This new method makes use of the state-of-the-art shrunk2D flow and its clock tree for clustering and relocates the flip-flops based on clock skew, clock tree hierarchy, and critical path information.

II. State-of-the-Art M3D Clock Router

Even though 3D ICs show potential for continuing Moore's law, there is currently no existing commercial 3D EDA tool that can be used to design a 3D chip. For this reason, we implemented our M3D designs using the state-of-the-art Shrunk2D flow [7], which makes use of 2D commercial EDA tools for the design of 3D ICs. Before the advent of the shrunk2D-flow, the traditional approach for routing clock trees in 3D ICs has been to construct a separate clock tree per tier and connect them through a single via. This means that the clock router will ignore the clock tree of the whole design and instead create a new one for each tier after logic partitioning. Unlike this conventional approach, the shrunk2D flow utilizes the resulting shrunk2D clock tree instead of abandoning it.

In the shrunk2D flow, shown in Fig. 1, cells and metal layer pitches are first scaled down by 0.707 and placed in a halved footprint. This allows commercial 2D tools to be used for PnR which will give the optimized x and y location of the cells. After PnR, the cells and wires is transformed back to their original sizes, which results in placement overlaps between cells. These overlaps of cells are resolved through tier partitioning using the Fidduccia and Mattheyses (FM) mincut partitioning algorithm [4] to decrease the connections between two dies. During this process, the clock tree is also partitioned. The backbone of the shrunk2D clock tree (clock buffers and clock gates) are fixed to a single tier and the flip flops are partitioned into either tier using area balance. Since MIVs are small and consume less power compared to the long wires that was necessary in 2D ICs, the flow has no constraint in adding MIVs. With the assistance of MIVs, the flow utilizes the shrunk2D clock tree instead of building new ones for each tier. The resulting M3D design based on this shrunk2D design flow will be used as the baseline for comparison.

This research was supported by the DARPA ERI 3DSOC Program under Award HR001118C0096, grant from Taiwan Semiconductor Manufacturing Company, and the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2016R1D1A1B03932711.

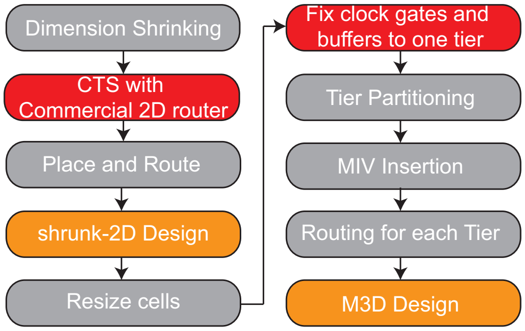


Fig. 1: **Baseline Shrunk2D [7] flow: CTS steps are in red**

Although the shrunk2D flow aims for minimum variation to the shrunk2D solution, during the process of resolving overlaps and partitioning logic, the x, y location of each cell cannot remain completely the same. Thus, the clock tree also needs modification to adjust for the changes to provide better results. With the clock related parameters considered, the quality of the M3D design can be improved further. There have been efforts to build better RTL-to-GDSII design flows for M3D ICs such as the cascade2D flow which utilizes design and micro-architecture insight and the compact2D flow that does not require geometry shrinking but uses scaled interconnect RC parasitics [1][6]. There have also been efforts to improve the shrunk2D flow itself by applying parasitic adjustments for more accurate estimates of RC parasitics of wires in M3D design [2]. Although this technique provides power savings, it requires the modification of RC parasitic lookup tables which are often encrypted in proprietary PDKs.

Instead of coming up with an entirely new flow, we modified the shrunk2D flow to include our newly proposed algorithm that does not require any additional modification of proprietary information. The new algorithm accounts for the hierarchy of the clock tree as well as the final timing of the M3D design (slack), clock skew, and flip-flop location.

III. M3D Clock Tree Optimization

A. Clock Tree Clustering Method

The modified shrunk2D flow with the clock tree optimization steps is shown in Fig. 2. Starting with a shrunk2D design, the x and y location of each clock cell, as well as the clock tree hierarchy of the shrunk2D design, is extracted. Using the topological clock tree (refer to Fig. 7) information, we cluster the buffers. With the shrunk2D design, an initial M3D design will be implemented for purposes of extracting the top critical paths. Using both the critical path information and topological clustering information, flip-flops will be relocated so that they will be closer to the center of each cluster. Once flip-flops are relocated, tier partitioning is performed using FM mincut partitioning with the exception of the clock tree. For the clock tree, partitioning of the clock cells per cluster is performed. Finally each tier is routed and a new M3D design with a new clock tree is obtained.

B. Tier Partitioning Method

In the original shrunk2D flow, tier partitioning is done using FM mincut partitioning to minimize the connections between all cells on the top and bottom tier. Our algorithm utilizes this information from the shrunk2D flow and partitions the flip-flops onto the majority tier. For example if there are 5 flip-flops in a cluster and only 1 of them is on the top tier while the others are on the bottom, all of them will be fixed to the bottom tier. This will remove MIVs on the clock path which could have caused extra clock skew within that cluster. Fig. 4 shows this concept.

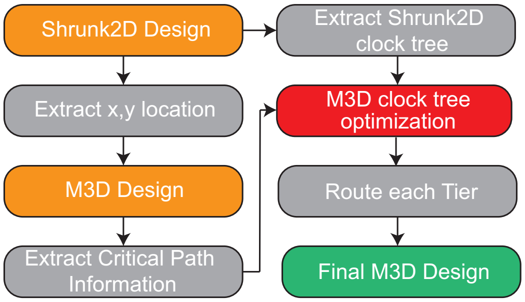

Shrunk2D Design
Extract Shrunk2D clock tree
Extract x,y location
M3D clock tree optimization
M3D Design
Route each Tier
Extract Critical Path Information
Final M3D Design

Fig. 2: **New flow for our M3D clock tree optimization.**

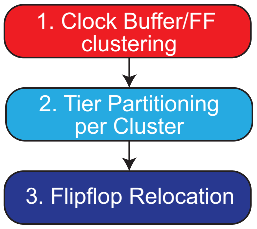

1. Clock Buffer/FF clustering
2. Tier Partitioning per Cluster
3. Flipflop Relocation

Fig. 3: **Three steps involved with our M3D clock tree optimization. Some steps can be skipped if desired.**

C. Flip-Flop Relocation Method

High clock skew among close-by cells can potentially cause timing violations or addition of buffers during PnR to close timing which will end up consuming more power. In the new flow, flip-flops are moved closer to each cluster center depending on the clock skew average in order to achieve a smaller clock skew. With flip-flop relocation, local clock skew is reduced. For example, with the flip-flop pair shown in Fig. 5, if FF2 is moved relatively closer to FF1 and the network of buffers, the connecting wire for both the logic path and the clock path becomes shorter. With a shorter connecting wire, both the logic path delay and the clock path delay are decreased which also results in less clock skew.

However, relocation of flip-flops, must be handled with care as it may hurt a critical path or degrade the quality of a design. Moving a flip-flop closer to the cluster center in order to reduce delay and clock skew in one flip-flop pair could hurt another path that one of the flip-flops is part of. Thus, we move the flip-flops only if its clock skew is larger than the skew average and if it is on the top 50 critical paths. Fig. 6 shows how the flip-flop relocation works.

D. Slew Target Consideration

In order to find the parameters that are most relevant to the clock tree, 400 implementations of the same benchmark

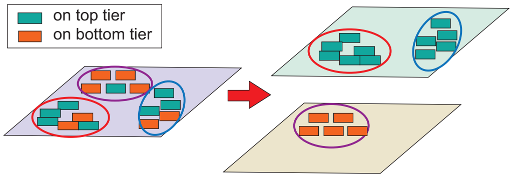


Fig. 4: **Tier Partitioning: Flip-flops are partitioned into top and bottom tier depending on which tier the majority of the flip-flops in the cluster were partitioned into during the previous iteration (left) with FM mincut partitioning.**

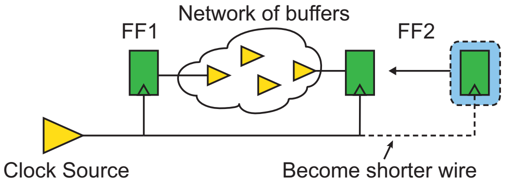


Fig. 5: **Benefits from flip-flop relocation: Moving a flip-flop closer to the cluster center shortens a wire connection which improves clock skew.**

with randomly picked parameter values were designed. The correlation between each parameter and the sequential power and clock combinational power was found. Table I shows the correlation values for each variable with sequential power and Table II shows the correlation values for each variable with combinational power. From Table I, we see that sequential power has the highest correlation with the input max transition of the leaf cells, which means the target slew has the highest effect. Thus, target slew is also a considered variable that is changed throughout the experiments.

E. Buffer Reduction

From Table II, we see that the buffer count has the highest correlation with clock combinational power. Unlike 2D ICs, M3D has shorter physical connections between cells on the top and bottom tier. Thus, it is easier to close timing with fewer buffers to obtain the same frequency as 2D. In order to reduce the number of buffers inserted during the design, we design M3D initially with a lower target frequency. This will further reduce the clock combinational power of M3D while still meeting timing.

F. Algorithm Description

As target slew is closely related to sequential power, the best slew target is decided first by multiple M3D implementations of the same benchmark. By sweeping with a couple slew targets, we find the slew that gives the best power results. Then, a new shrunk2D design is designed with the optimized slew target and buffer reduction. With the resulting shrunk2D design, the new algorithm performs topological clustering with shrunk2D clock tree. As shown in Fig. 7, the extracted clock tree from the shrunk2D design is clustered based on the level of the cell within the tree. For example, if the optimized level for a specific benchmark is level 4, all the clock trunks and leaves below each level 4 clock cells will be grouped together.

Algorithm 1 shows the whole flow from the topological clustering stage up to when tier partitioning and flip-flop relocation is performed with the cluster information.

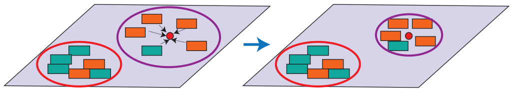


Fig. 6: **Flip-Flop Relocation: Flip-flops are moved closer to the cluster center if the skew is larger than the average skew of the design.**

TABLE I: **Correlation of variables with sequential power**

	Variable	Correlation
Sequential Power	Worst rising trunk slew	0.006
Sequential Power	worst rising leaf slew	0.256
Sequential Power	max skew	0.069
Sequential Power	CTS_MaxTran_trunk	-0.036
Sequential Power	CTS_MaxTran_leaf	0.566
Sequential Power	buffer count	-0.211

G. Parameters Under Optimization

Sweeping was done for the following parameters to find the optimal value for each parameter that yields the most power savings and PDP benefits for each benchmark:

- Clock tree hierarchy: Since the sequential cells are clustered based on the shrunk2D clock tree, the hierarchy of the clock tree is strongly reflected in the newly found clock tree solution. For each benchmark, the best level of the clock tree at which a new cluster is formed is different.
- Slack: Relocating flip-flops can potentially hurt a critical path and further degrade the worst negative slack. This can hurt the performance of the whole chip and needs to be considered carefully. Therefore, the flip-flop relocation happens only if the flip-flop is in the top 'n' number of paths. The number of paths (n) considered in this case also differs depending on the benchmark.
- Clock skew: Clock skew can cause delays throughout different paths in the circuit. Thus, a smaller skew among local flip-flops is desired. If the clock skew is larger than a percentage of the average of all the clock skews in the chip, the cell will be relocated to a closer location to the center of each cluster. A couple values for the percentage of the clock skew average was tested.
- Displacement: Too much displacement of a flip-flop can potentially harm a path and overall hurt the performance of the chip. Thus, different percentages of relocations have been tested. In algorithm 1, the cells are moved by one third of the distance between the cell itself and the cluster center. This percentage may differ per benchmark and was tuned for the best result in each of them.

H. Effect On Runtime

In a typical ASIC design, clock cells constitute about 10 to 20% total cell count. The delivery of clock signals to flip-flops is achieved by the clock tree where each node is split to multiple nodes. Since the clock tree created by the APR tool will be approximately balanced, we will assume a balanced k-ary tree for calculation purposes. Thus, the time complexity of sweeping through clock tree levels would be $log_k(0.2 * n) + c$ depending on the percentage of clock cells, where n is the total number of cells in the benchmark, and c is a constant added

TABLE II: **Correlation of variables with clock combinational power**

	Variable	Correlation
Clock Comb. Power	worst rising trunk slew	0.135
Clock Comb. Power	worst rising leaf slew	-0.089
Clock Comb. Power	max skew	0.256
Clock Comb. Power	CTS_MaxTran_trunk	0.147
Clock Comb. Power	CTS_MaxTran_leaf	-0.198
Clock Comb. Power	buffer count	0.656
Clock Comb. Power	sequential power	-0.439

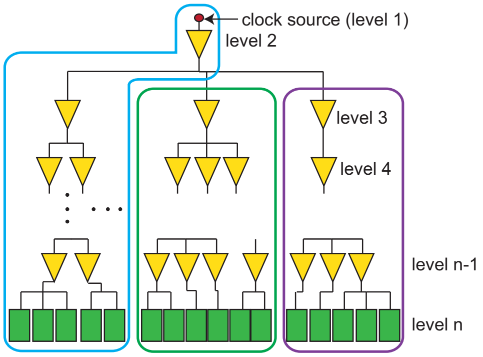


Fig. 7: **Clustering of clock tree using topology: We show clock tree clustering at level 2.**

for the unbalanced part of the tree. For example, in the case of a design with 10,000 cells using a 4-ary tree only needs 7 levels. In a more realistic clock tree the k is not a constant for every node and is usually high for the last level of supply and so our assumption is very conservative. For example, in the case of VGA, there are 54,489 cells but the depth of the clock tree is only 8. In run time, it only takes about 10-30 seconds to run algorithm 1 depending on the size of the benchmark.

The other parameters considered does not add to the time complexity as there are not too many values to be swept. For example, in the case of the displacement parameter, we have tried moving the clock cell by one third, one half, and two thirds of the distance between the cell and the cluster center. This is only three cases and thus should be considered a constant in terms of time complexity. Similarly, the clock skew parameter and the number of paths we consider for slack should be considered a constant for time complexity as well.

IV. Experimental Results

A. Experimental Setup

A commercial 28nm PDK was used for both the baseline M3D and new algorithm based M3D designs. The algorithm was tested on 4 different benchmarks from OpenCores.org: VGA Controller, AES-128, AVC, and the JPEG Encoder. The shrunk2D flow based M3D designs were used for baseline comparison. A reasonable target frequency for each benchmark was chosen and used for implementation of 2D, baseline M3D, and our algorithm based M3D designs. All other design parameters such as the size of the footprint and the number of metal layers used are kept constant for all designs. For better visualization of the effect of clocktree clustering and flip-flop relocation, benchmarks with comparatively high sequential cell percentages have been chosen. The sequential cell count and the percentage of sequential cells of each benchmark are shown in Table III. The benchmark VGA shows the highest percentage of sequential cells.

```
Topological Clustering: ;
for cell in clock tree do
    if clock cell is target level in clock tree then
        create new cluster
    end
end
Find average of clock skew of all cells in clock tree ;
Flip-Flop Relocation : ;
for cell in cluster do
    if clock skew $>$ clock skew average then
        if cell is in top 50 critical paths then
            Xlocation = Xlocation + (clusterCenterX -
              Xlocation)/3 Ylocation = Ylocation +
              (clusterCenterY - Ylocation)/3
        end
    end
end
Tier Partitioning with Cluster Result: ;
for each cluster do
    if number of cells in cluster on top die $>$ number of cells in
      cluster on bottom die then
        move all cells in cluster to top die
    end
    else if number of cells in cluster on top die $<$ number of
      cells in cluster on bottom die then
        move all cells in cluster to bottom die
    else
        move all cells in cluster depending on entire cell
          balance
    end
end
```

ALGORITHM 1: Our M3D clock tree optimization

TABLE III: **Sequential cell percentage of each benchmark**

	VGA	AES-128	TATE	JPEG
Seq. Cell Count	17,055	10,688	31,416	37,665
Cell Count	54,489	101,957	210,220	227,406
Percentage	**31.5**	10.5	14.9	16.6

B. Clock Tree Optimization Results

The effect of our new tier partitioning and flip-flop relocation algorithm on clock tree quality and total power is shown in Table IV. Three methods are compared in the table: tier partitioning only, flip-flop relocation only, and the result from performing both methods at the same time. All designs for a single benchmark were compared at iso-performance and the last row for each benchmark compares the best method out of the three to the baseline M3D design. All 4 benchmarks show great savings in clock tree wirelength, clock skew, and buffer count. Overall, the benchmark AES showed the most improvement in clock tree quality with a 35.9 % saving in clock tree wirelength and 34.3 % less clock skew. This also leads to a 15.5 % saving in total power compared to the baseline.

VGA clock tree quality is improved using both methods as shown in Table IV. With both methods performed, we achieve up to 28.3% clock skew saving and 33.6% better latency compared to the baseline M3D. With our algorithm, the clock tree wirelength is also reduced by 30.8%. The number of buffers increased slightly in order to meet the optimized slew target, but with the addition of 1.8% buffers, the internal power

TABLE IV: **Clock Tree Quality Comparison. A negative Δ % value means improvement.**

		Freq	WNS	Seq pwr	Clk Comb. pwr	Total pwr	clk WL	clk skew	clk latency	# Buffer
	Options	MHz	ps	mW	mW	mW	m	ps	ps	-
VGA	baseline [7]	2,250	93.4	203.2	34.9	276.1	0.0172	104.9	202.4	6,223
	tier partitioning	2,250	59.2	180.3	36.3	254.8	0.0119	86.4	144.6	6,336
	FF relocation	2,250	-20.3	180.4	40.6	258.0	0.0193	81.7	149.3	6,336
	both	2,250	-6.9	180.1	36.8	254.7	0.0119	75.2	134.4	6,336
	Δ %	-	-	-11.4	5.4	-7.8	-30.8	-28.3	-33.6	1.8
AES-128	baseline [7]	3,000	51.4	175.4	37.1	391.2	0.0167	57.5	165.7	22,269
	tier partitioning	3,000	44.0	170	33.4	330.7	0.0107	37.8	143.6	17,200
	FF relocation	3,000	31.5	170.5	34.92	332.3	0.0158	39.9	156.5	17,200
	both	3,000	56.8	173	34.2	336.2	0.0118	46.4	158.5	17,200
	Δ %	-	-	-3.1	-10.0	-15.5	-35.9	-34.3	-13.3	-22.8
TATE	baseline [7]	2,000	124.1	317.4	63.0	602.7	0.0391	53.9	189.1	18,297
	tier partitioning	2,000	96.3	308.1	62.6	565.7	0.0269	35.7	181.4	17,522
	FF relocation	2,000	73.3	308.2	66.8	568.4	0.0401	54.4	192.3	17,522
	both	2,000	-47.8	310.1	63.5	573.4	0.0277	45.9	191.3	17,522
	Δ %	-	-	-2.9	-0.6	-6.1	-31.2	-33.7	-4.1	-4.2
JPEG	baseline [7]	1,600	91.7	303.5	69.5	811.6	0.0647	180.5	339.5	29,604
	tier partitioning	1,600	90.6	297.5	65.7	773.2	0.0438	145.2	298	24,686
	FF relocation	1,600	57.3	297	72.3	775.9	0.0674	152.9	299.9	24,686
	both	1,600	65.2	297.6	67.8	776.2	0.0467	163.4	308.6	24,686
	Δ %	-	-	-1.9	-5.5	-4.7	-32.3	-19.6	-12.2	-16.6

decreased and thus the overall total power decreased. With the optimized parameters, a better clock tree is obtained overall and a the design also consumes less power. Although there is a small WNS of -6.9 ps, this does not affect the operating frequency by too much.

AES-128 clock tree quality is best when only tier partitioning is applied. Flip-flop relocation does not have much effect in the case of this specific benchmark. With tier partitioning only, max clock skew and the clock tree wirelength are improved significantly from the design using the original M3D design flow showing -34.3% less clock skew and 35.9% clock tree wirelength saving. With the healthy amount of clocktree wirelength saving and 22.8% buffer reduction, we achieve 15.5% reduction in total power as well. The clock trees for AES with all four methods are shown in Fig. 8. Flip-flop relocation did not change the clock tree dramatically, but tier partitioning made a visible change in the clock tree. This is because flip-flop relocation in the x-y plane can be risky and is moved only when the conditions described in section 3.3 are met. With tier partitioning, we see that there are few locations where flip flops are on both the top and bottom tiers.

TATE does not show too much clock combinational power or sequential power savings but with a better clock tree, it shows up to 6.1% total power savings compared to the baseline M3D design. JPEG shows the least amount of power savings with only 4.7% power reduction. Although the power did not reduce significantly, the overall quality of the clock tree is improved with 32.3% less clock tree wirelength and 5.5% saving in clock combination power.

Fig. 9 show the clock trees with sequential cells and clock tree routing for the benchmark VGA. The left shows the original M3D design and the right shows the new design completed with the new algorithm. In this case, performing both tier partitioning and flip-flop relocation showed the best results and thus the clock tree shown on the right is the result of both methods. The best results for VGA are found when most of the clock tree is moved to the bottom die.

TABLE V: **Comparison of our M3D clock tree optimizer with commercial 2D (Cadence Innovus) clock router. A negative Δ % value means improvement.**

	Metric	2D	Best M3D	Δ %
VGA	Clock Skew (ps)	81.1	75.2	-7.2
	Clock Latency (ps)	177.4	134.4	-24.2
	Total Power (mW)	281.9	254.7	-9.6
AES-128	Clock Skew	43.6	37.8	-13.3
	Clock Latency	189.6	143.6	-24.3
	Total Power	411.7	330.7	-19.7
TATE	Clock Skew	46.4	35.7	-23.1
	Clock Latency	223.3	181.4	-18.8
	Total Power	623.7	565.7	-9.3
JPEG	Clock Skew	180.4	145.2	-19.5
	Clock Latency	347.6	298	-14.3
	Total Power	851.3	773.2	-9.1

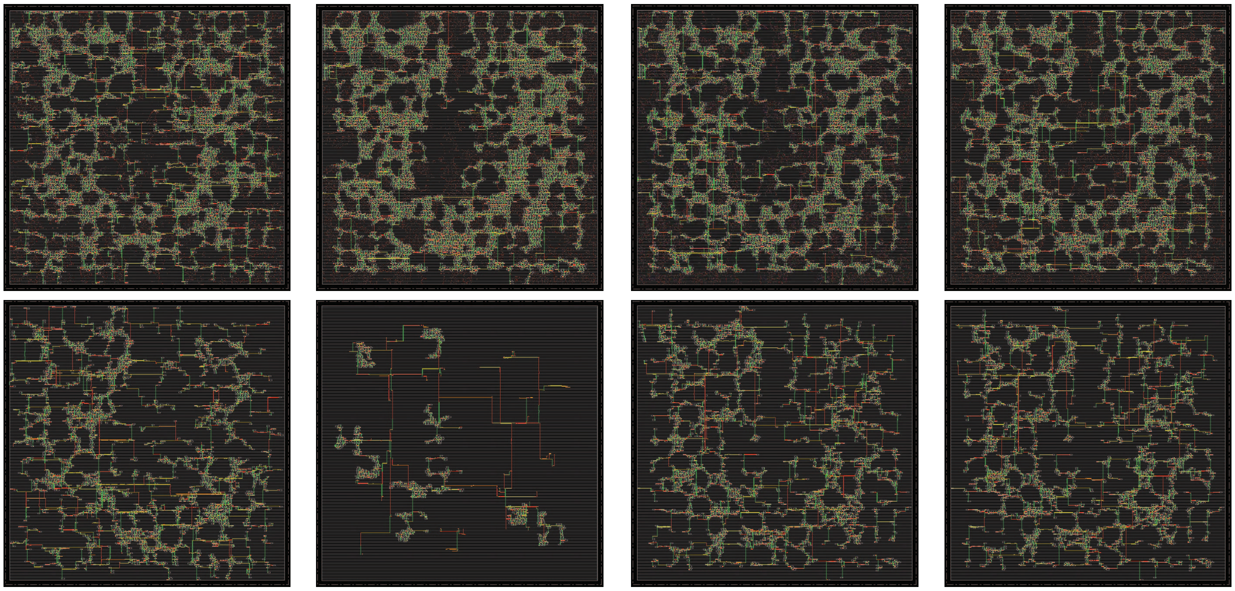
C. Comparison with Commercial 2D PNR Tool

The best M3D results are also compared to their 2D counterparts in Table V. AES shows the most power savings compared to 2D with 19.7% total power savings. VGA and TATE each show 11.9% and 9.3% savings respectively. JPEG shows the smallest power savings with 8.9% reduction in total power compared to 2D. The clock skew and latencies are also compared in the table. Although there are some outliers in the trend, overall, M3D designs show better clock skew and clock latency compared to the 2D designs.

D. Summary

Observations from the clock tree clustering methods are summarized below:

- Our method shows better power and clock skew compared to the baseline M3D design.
- Power savings from buffer reduction is larger at higher target frequencies because more effort is necessary to close timing in 2D IC designs.
- For M3D, reducing buffer count by initially designing with a lower target frequency is a valid design decision. With vertical connections, timing is still met unlike in 2D.



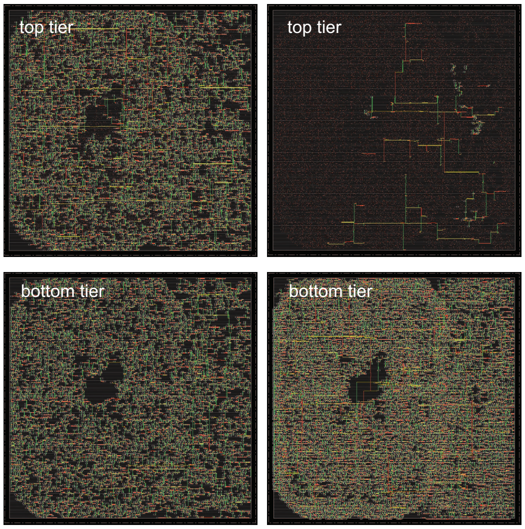
(a) AES original

(b) tier partitioning only

(c) Flip-flop relocation only

(d) Both combined

Fig. 8: **Clock tree comparison for AES. Tier partitioning method show the best results with 15.5% total power saving.**



VGA original

VGA with partitioning and FF relocation

Fig. 9: **Clock tree comparison for VGA. The clock tree on the right has 29.6% better clock latency.**

- Different benchmarks show better results with different methods, e.g using both tier partitioning and flip-flop relocation works best for VGA, but only performing tier partitioning works best for AES-128.
- Clock tree tier partitioning is effective in reducing clock skew and clock combinational power.
- With change in target slew, further improvement in clock latency and sequential power can be achieved.
- A tighter clock slew target does not always result in sequential power saving and thus the optimized slew target must be found to achieve the best results.

V. Conclusions

In this paper, we explored clock tree optimization methodologies and their impact on power. Main ideas include topological clock tree clustering and using the clock tree hierarchy for tier partitioning and flip-flop relocation. Slew considerations and buffer reduction through lower target frequency further improve total power. We re-confirmed the effect of clock tree quality on full chip power and provided various methodologies to improve clock tree and full chip power for M3D designs. For further work, machine learning could be used to find the optimal parameters since optimizing the parameters considered in our algorithm take time and effort.

References

[1] K. Chang et al. Cascade2D: A design-aware partitioning approach to monolithic 3D IC with 2D commercial tools. In *Proc. IEEE Int. Conf. on Computer-Aided Design*, 2016.

[2] K. Chang, S. Pentapati, D. E. Shim, and S. K. Lim. Road to high-performance 3d ics: Performance optimization methodologies for monolithic 3d ics. In *Proceedings of the International Symposium on Low Power Electronics and Design*, ISLPED '18. ACM, 2018.

[3] G. W. et al. Flip-flop clustering by weighted k-means algorithm. In *2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6, June 2016.

[4] C. M. Fiduccia and R. M. Mattheyses. A linear-time heuristic for improving network partitions. In *19th Design Automation Conference*, June 1982.

[5] A. B. Kahng, J. Li, and L. Wang. Improved flop tray-based design implementation for power reduction. In *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, Nov 2016.

[6] B. W. Ku, K. Chang, and S. K. Lim. Compact-2d: A physical design methodology to build commercial-quality face-to-face-bonded 3d ics. In *Proceedings of the 2018 International Symposium on Physical Design*, ISPD '18. ACM, 2018.

[7] S. Panth, K. Samadi, Y. Du, and S. K. Lim. Design and cad methodologies for low power gate-level monolithic 3d ics. In *2014 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, Aug 2014.

[8] S. A. Panth, K. Samadi, Y. Du, and S. K. Lim. Shrunk-2D: A Physical Design Methodology to Build Commercial-Quality Monolithic 3D ICs. *IEEE Trans. on Computer-Aided Design of Int. Circuits and Systems*, 2017.

[9] X. Zhao and S. K. Lim. Through-silicon-via-induced obstacle-aware clock tree synthesis for 3d ics. In *17th Asia and South Pacific Design Automation Conference*, 2012.