Placement Aware Clock Gate Cloning and Redistribution Methodology

Abstract

Gating clocks has been a widely adopted technique for reducing dynamic power. The clock gating strategy employed has a huge bearing on the clock tree synthesis quality along with the impact to leakage and dynamic power. This paper proposes a technique for clock gate optimization to aid clock tree synthesis. The technique enables cloning and redistribution of the fanout among the existing equivalent clock gates. The technique is placement aware and hence reduces overall clock wire length and area. The method involves employing the "k-means clustering algorithm" to geographically partition the design's registers. This enables better clock tree quality entitlement during clock tree synthesis in terms of clock tree area, power and better local skew distribution. The paper highlights the utility of this technique by showcasing the clock tree synthesis quality of results improvement on a complex design.

1 Introduction

Shutting off the clock to a register or a bank of registers when it is not known to change state is a common but effective practice to reduce dynamic power [1]. But there are a lot of design considerations that crop up during the actual implementation of the clock gates. The following are a few critical aspects that need to be considered while coming up with the clock gating strategy.

1. Dynamic power and leakage power tradeoff.
2. Tradeoff between dynamic power savings and setup timing closure to the enable pins.
3. Effect of clock gating on the clock tree divergence.

Each of these considerations has a huge impact on the quality of the clock tree in terms of insertion delay, area and divergence. The positioning of the clock gate with respect to the flops in its fanout is very critical is a very important aspect of handling clock gates. The placement of flops is typically driven by their connectivity and timing characteristics [3]. In such a scenario, it is extremely important to have grouping of the flops under the clock gates based on their physical proximity to ensure lower divergence and local skew. This paper proposes a physical placement aware technique for cloning and redistribution of clock fanout among equivalent clock gates. The paper presents the commonly used methods of handling clock gates in physical design that are in practice along with their clock tree synthesis care-abouts. The next section describes in detail the problem statement the paper is trying to address. The fourth and fifth sections respectively describe how the "k-means clustering algorithm" has been adapted to clone clock gates and redistribute the fanout among equivalent clock gates for better clock tree synthesis quality. Subsequently, the paper showcases the clock tree synthesis quality improvements observed with this technique when tried on a complex high speed processes sub system design. The paper concludes with the future work that this work could extend to.

2 Current Practices

In any digital synchronous design, disabling the clock signal to the registers when they are not in use reduces the active power of the circuit. This clock gating is either implemented in RTL by the designer with his knowledge of the design's activity. Also, in cases when the data to a register is gated by an enable signal, alternatively the enable signal could be used to gate the clock itself to the register there by reducing the active power. Fig.1 illustrates this idea of converting data gating to clock gating.

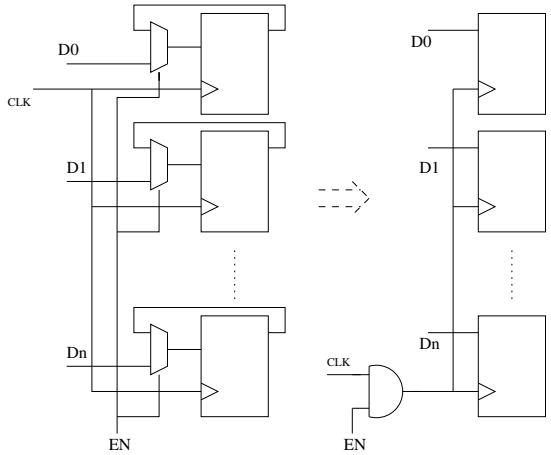

Fig.1 EN signal used to gate the data can be used to gate the clock to reduce active power.

There are EDA solutions today to identify the data gating scenarios in the RTL and automatically convert them into clock gating circuitry during synthesis of the design itself. One of the aspects that need to be considered while introducing clock gates automatically is the leakage power cost of the clock gates. The leakage power of the clock gate added should not exceed the dynamic power savings the clock gate brings. Hence it has to be ensured that a clock gate inserted should be gating off a minimum set of registers so that a significant active power saving can be achieved. The higher the fanout of the clock gate, more the dynamic power saved. But with high fanout, a buffer tree at the output of the clock gate would be necessary to efficiently drive the high fanout. This makes the insertion delay to the clock gate much lesser than that of the registers. The lower clock latency to the clock gate could potentially translate to difficulties in meeting setup timing on the enable signals of the clock gates. It is thus important to limit the fanout of the clock gates by cloning them to ensure predictable timing closure. Placement of the newly inserted clock gates is a critical care about. Suboptimal placed clock gates could lead to increase in the clock tree area and insertion delay when the clock tree is synthesized subsequently. The resulting clock tree would also have more divergence and thus making it vulnerable to on chip variation effects. A commonly used methodology for clock gate insertion to account for the above care-abouts involves:
1. Inserting clock gates only if it can gate the clock to a minimum set of registers.
2. Set a max limit on the fanout of a newly inserted clock gates. This would lead to creating more clones of the clock gate. Typically, clock gating is done on the RTL itself and the partitioning of the registers to the various clones is done heuristically as placement data is not available at that point. This method, not being placement aware, can cause sub optimal clock gating during layout. Disparate placement of registers of a common bank can limit clock tree synthesis quality of results as illustrated before. There are some EDA solutions to handle the cloning of clock gates during to layout implementation to address the above issue. But these solutions are more focused on the enable timing issues and often cause clock tree synthesis quality issues like blow up in clock gate area and clock insertion delay.

3 Problem Statement

This paper presents an improved cloning methodology that is physical placement aware to ensure good clock tree synthesis (CTS) quality of results(QoR). The paper also proposes a clock gate fanout redistribution technique based on physical proximity of the registers for previously cloned designs.

4 Clock Gate Cloning

4.1 Proposed Clock Gating Scheme

A simple clock gate insertion methodology is shown in Fig.2

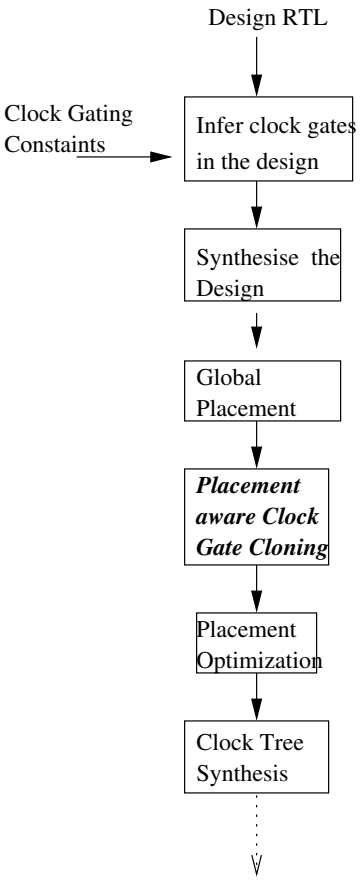

Fig.2 Clock gate insertion scheme

This methodology involves inserting clock gates in RTL with a suitable minimum fanout limit constraint but with no upper bound on the fanout of the clock gates. During layout implementation, when the placement data is available, the clock gates can be cloned using the proposed algorithm. This makes the solution fully aware of placement and ensures good CTS QoR subsequently.

4.2 Placement Aware Cloning Algorithm

Cloning of a clock gate involves creating multiple equivalent clock gates and distribution of the fanout of the clock gate among the newly created clock gates. The proposed technique identifies clock gates for cloning if it satisfies any of the following criterions:
i. Fanout of the clock gate is higher than a upper bound.
ii. If the fanout of the clock gate is spread over a large area.

Once the clock gates to be cloned are identified, the clones are created and the fanout of the parent clock gate is partitioned geographically and assigned to the clock gate and its clones. The technique employs "k-means algorithm" [2] to partition the registers.

4.2.1 K-means Clustering Algorithm

K-means clustering is a method of cluster analysis which aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean. It iteratively refines the clustering and the means to arrive at the cluster partition. Given a set of observations $x_1, x_2, ..., x_n$, where each observation is a d-dimensional real vector, k-means clustering aims to partition the n observations into k sets ($k \leq n$) S = $S_1, S_2, ..., S_k$ so as to minimize the within-cluster sum of squares (WCSS):

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in S_i} \|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2$$

where μ_i is the mean of points in S_i. Given an initial set of k means $m_1, m_2, ...m_k$, the algorithm converges on the partitions by alternating b/w the following 2 steps:
1.Assignment step: Assign each observation to the cluster with the closest mean (i.e. partition the observations according to the Voronoi diagram generated by the means).

$$S_i^{(t)} = \left\{ \mathbf{x}_j : \|\mathbf{x}_j - \mathbf{m}_i^{(t)}\| \leq \|\mathbf{x}_j - \mathbf{m}_{i^*}^{(t)}\| \text{ for all } i^* = 1, \ldots, k \right\}$$

where ***t*** is the iteration number.
2.Update step: Calculate the new means to be the centroid of the observations in the cluster.

$$\mathbf{m}_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{\mathbf{x}_j \in S_i^{(t)}} \mathbf{x}_j$$

The algorithm is deemed to have converged when the assignments no longer change.

4.2.2 K-means Clustering Adaptation for Clock Gate Cloning

The flowchart in Fig.3 describes how the standard k-means algorithm is adopted for partitioning flops under the clock gates/clones.

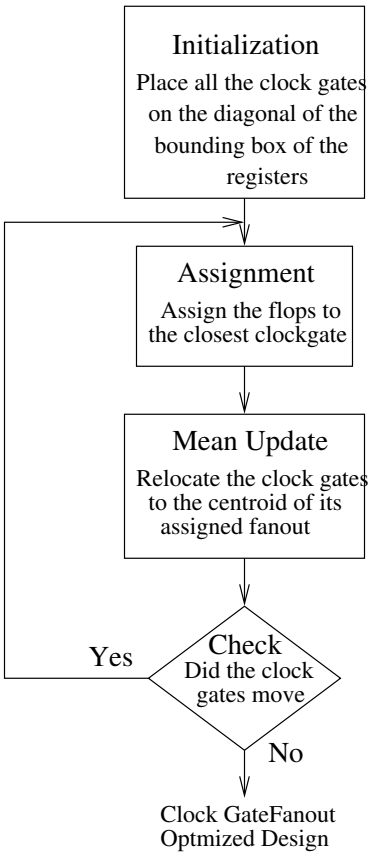

Fig.3 Proposed clock gate cloning methodology.

Initially the clock gate and its clones are placed on the diagonal of the smallest rectangle containing all the registers in the fanout of the parent clock gate. This forms the initial locations of the means for the algorithm. Each register is then attached to the nearest clock gate. In case the clock gate has already reached its fanout limit, it is attached to next nearest clock gate. Once all the registers are assigned to the clock gates, the location of the clock gates are recalculated as the mean location of the registers it is assigned. The register assignment and clock gate location relocation steps are repeated iteratively to obtain the best physical partition of the registers. The iterations can be stopped as soon as the following 2 criterions are satisfied.
1) The registers are not getting reassigned any more.
2) The locations of the clock gates are not changing anymore.
This is done for every clock gate that has been identified for cloning. This technique ensures that all the clock gates are driving registers that are clustered together on the layout and also that the clock gate is placed exactly at the load center of its fanout, These make the clock gating structure very conducive for good CTS QoR entitlement.
Fig.4 illustrates the results of placement aware cloning on a sample design. The single clock gate driving all the flops is efficiently cloned to handle smaller and more physically

localized collection of flops.

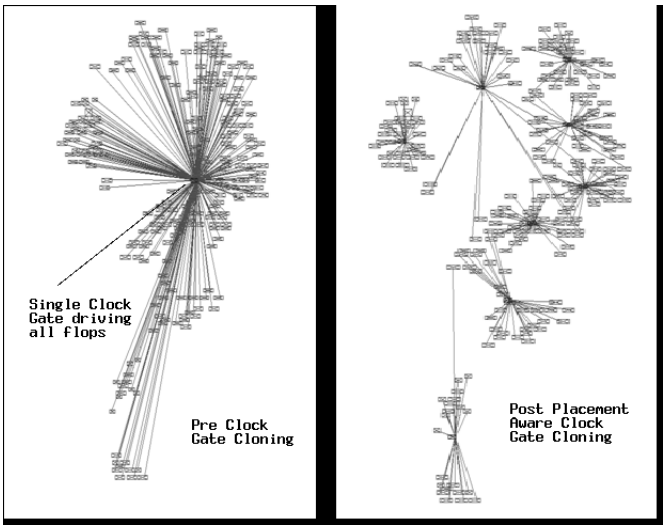

Fig.4 Cloning on a sample design.

5 Redistribution of Clock Gate Fanout

In cases when the design already has the clock gates cloned during the insertion of clock gates, the technique can be used to find equivalent clock gates and redistribute the fanout among them. All clock gates driven by the same set of control signal are considered equivalent. Then the aforesaid technique for partitioning the registers and refining the clock gate placements using the k-means algorithm can be employed.

6 Results

The proposed methodology was used for optimizing the clock gating on a 45nm processor subsystem and the results were bench marked against the other solutions. The run time to clone a little over thousand clock gates was less than 15 minutes when executed on a Linux box. The memory overhead of the algorithm is insignificant. Table1. shows the clock tree synthesis results with the various options of handling clock gates.

	Without any Clock gate optimization	Clock gates cloned with an EDA solution	Clock Gates cloned using the proposed technique
Max Insertion Delay	946ps	928ps	974ps
Skew	71ps	61ps	63ps
Clock Tree Buffer Area	2801.9u2	764.1u2	558.3u2
Clock Gate Area	7010.6u2	12729.2u2	6715.4u2
Total Clock Tree Area	9812.5u2	13493.4u2	**7273.7u2**

Table.1 Clock tree synthesis results with various clock gate cloning options.

The results clearly highlight the benefit of the optimized cloning solution. It is evident that the cloning has improved the local skew of the design which directly enhances the performance of the design. The more important benefit is seen with respect to the clock tree area. There is a 25% reduction in the clock tree area that the proposed cloning solution brings as against to not cloning. Also, the area benefit obtained when compared to a commercially available EDA cloning solution is much higher. This area benefit directly implies lower static and active power consumption of the clock tree.

7 Conclusion and Future Work:

The results highlight the value the technique brings into the design in terms of clock tree area. This in turn reduces the leakage power of the design. In addition to the area improvement, the methodology also sets up the design for implementing relative placements for the flops and clock gates. The relative placement is a commonly used practice to reduce leaf clock power and also area reduction. Using the proposed method ensures that the regular placement implementation can be implemented without big register displacements. As a future work to this effort, we need to develop an efficient methodology to declone clock gates. In scenarios where we already have a design with clock gates already cloned, merging equivalent clock gates to optimize area and also satisfy all the above requirements mentioned in the paper earlier.

References

[1] L. Benini, P. Siegel, and G. D. Micheli. Fautomated synthesis of gated clocks for power reduction in sequential circuits. *IEEE Design and Test of Computers*, pages 32–41, 1994.

[2] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. An efficient k-means clustering algorithm : Analysis and implementation. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 7:881, July 2002.

[3] R. V. Raj, N. S. Murty, P. S. N. Rao, and L. M. Patnaik. Effective heuristics for timing driven constructive placement. *VLSI Design*, pages 38–43, 1997.