ISSN 23499842(Online), Volume 1, Special Issue 2(ICITET 15), March 2015
International Journal of Innovative Trends and Emerging Technologies

LOW POWER CLOCK TREE SYNTHESIS USING CLUSTERING ALGORITHM

P.SELVANAND[1], U.RAGAVENDRAN[2]
[1]PG Student, VLSI Design, [2]Assistant Professor, [1,2]Dept of Electronics and Communication Engineering,
[1,2]SRM University, Kattankulathur, India

Abstract: Semiconductor technology scaling requires continuous evolution of all aspects of physical design of integrated circuits. Among the major design steps, clock tree synthesis has been greatly affected by technology scaling. Power is one of the complex growing issues of VLSI system. Clocks are known to be major source of power consumption in digital circuits. In Clock tree synthesis to create the clustering algorithm for a design to achieve the low power and build the better clock tree in terms of clock skew and latency. In this project a clustering algorithm for the minimization of power in a local clock tree is proposed. Given a set of sequential and their locations, clustering is performed to determine the clock buffers that are required to synchronize the sequential, where a cluster implies that a clock buffer drives all the sequential in the cluster. The results produced by the algorithm are often within lower bound and have lower costs, on average, than those due to an approximation algorithm with faster runtimes. The algorithm has been employed for clock tree synthesis for several microprocessor designs across process generations due to consistently significant clock tree power savings over the results due to competitive alternatives. The need for multi-objective optimization over a large parameter space and the increasing impact of process variation make clock network synthesis particularly challenging.

I. INTRODUCTION

Power dissipation has become a major concern for very large-scale integration designs. Today's microprocessors consume from a few 100mW to around 100W power. More than half of this power is dynamic. The clocks contribute major portion of that dynamic power. The distribution of the clocks for these circuits is performed employing global and local hierarchy. The global clock distribution employs a grid to distribute the clock from phase-locked loop to each block through several spines. The local distribution inside the block is performed by creating fixed stage buffered trees to facilitate the conditional gating of clocks to save the power. The dynamic power in local clocks can be attributed to:1) Interconnect capacitance of the clock nets.2) Gate capacitance of the sequential.3) Gate and diffusion capacitance of clock buffers. The interconnect capacitance contributes a major share of the power and motivates this paper on a power-aware clustering algorithm employed to perform the duplication of clock buffers during clock tree synthesis (CTS). In case of CTS, clustering approaches are employed for power minimization. The clustering technique in applies a heuristic based on dynamic programming with a metric from for wire length minimization during the clustering. To focuses on creating robust clock trees in the presence of process variability either by adding links at levels close to sinks or at higher levels, as in or by inserting tunable buffers that can be programmed after manufacturing. Toward the same goal, proposed adding extra trees between critical sink they also proposed placing registers, by considering the clock tree construction in the presence of blockages. The proposed approximation algorithms for a facility location problem with service capacities They have employed the algorithms to address the same clustering problem in CTS as in this paper. Other recent advances in CTS area include the use of multi bit flip-flops and the pulsed latches. The multi bit flip-flop creation from single-bit flip-flop design and replacement of those to safe power. The sequential in the design also consume significant power. To reduce the power consumed by those, pulsed latches can be employed instead of flip- flops. The clustering algorithm in this work with changes in constraints. They further extend the work to include co synthesis of the clock network and placement for pulse-latch designs to carry out the clock buffer duplication, we propose a greedy clustering algorithm aimed at minimizing power dissipation in the clock tree. The proposed algorithm also exhibits more than 2.5× runtime speedup. Across several microprocessor designs in different advanced nanometer process technologies, the algorithm has shown more than 10% reduction in clock tree power, on average, over the competitive alternatives consistently. The remainder of this paper is organized as follows. Section II describes the preliminaries. Section III defines the clustering problem. Section IV presents the architectural design flow compares Section V shows the Simulation results for our algorithm. Section VI concludes this paper.

II. PRELIMINARIES

The Clock tree synthesis (CTS) is a critical step in the physical implementation flow. An optimized clock tree (CT) can help avoid serious issues (excessive power consumption, routing congestion, elongated timing closure phase) .Clock gating structures, CTS targets, clock library cell types and even placement of spare cells have direct impact on the quality of a clock tree. Monitoring the QoR can become a challenge for CTS experiments that generate excessive amounts of data. Clock gating is a popular technique used in many synchronous circuits for reducing dynamic power dissipation. Clock gating saves power by adding more logic to a circuit to prune the clock tree. Pruning the clock disables portions of the circuitry so that the flip-flops in them do not have to switch states. Switching states consumes power. When not being switched, the switching power consumption goes to zero, and only leakage currents are incurred. The CTS distributes the global clock signal by creating a buffer tree according to the clock schedule and by obeying the constraints, such as maximum load, skew, wire delay on the clock nets, slope requirements, and so forth. It is mainly carried out in three steps: 1) the duplication of clock buffers; 2) the routing of the clock nets; and 3) the sizing of the clock buffers. The first step determines how many clock buffers will be created and which buffers will drive which receivers. A synchronous circuit consists of two kinds of elements: registers and combinational. Registers (usually implemented as D flip-flops) synchronize the circuit's operation to the edges of the clock signal, and are the only elements in the circuit that have memory properties. Combinational logic performs all the logical functions in the circuit and it typically consists of logic gates. A Zero Skew Clock Routing Methodology has been developed to help design team speed up their clock tree generation process. The methodology works by breaking up the clock net into smaller partitions, then inserting clock buffers to drive each portion, and lastly, routing the connection from original clock source to each newly inserted clock buffers with zero skew. A few Perl scripts and a new Visual Basic based routing tool have been developed to support the methodology implementation. The routing algorithm used in this tool is based on the Exact Zero Skew Routing Algorithm. For multiple stages, the duplication is performed recursively: it is first applied to k-stage sequential to determine k-stage buffers, which are then clustered to determine $(k-1)$-stage buffers, and so forth. We let the k, the number of stages, to be 1 for the sake of illustration in the remainder of this paper, except for Section V, where the results with actual number of multiple stages are reported.

III. PROBLEM FORMULATION

Given placement and sizes of sequential to be driven by one stage of clock buffers, the goal of CTS is to create the number of buffers such that each one drives the load within its strength limit, maintains the required slopes, and meets the delay or skew targets at all the receiving sequential. The power cost of the local clock tree running at the frequency f is given by PLCT = $a_1 + 1\ g__(S$equential + Interconnect)V^2f + Leakage $g_(S$equential + Interconnect) (1) where a, CLCT, V, $g\_$, and Leakage are the activity factor, the total capacitance of the clock tree, the supply voltage, the buffer gain, and a constant accounting for leakage in buffers, respectively. The details of the above derivation can be found in the CTS problem, only varying component is C Interconnect, and therefore, minimizing clock tree power is equivalent to minimizing interconnect capacitance, which can be controlled by the duplication based on clustering and by the routing. If a consistent routing scheme, for instance, zero skew one as in is assumed, then the interconnect capacitance is entirely dependent on the clustering. It is usually not obvious which clustering is good from the power perspective, since interconnect capacitance is known only after the routing. This motivates a need for metric that can guide the clustering with good fidelity. The minimum spanning tree (MST) proves to be such a metric suitable for greedy optimization , and therefore, we use the same. Using the equivalence between the minimization of the power in clock trees with that of the interconnect capacitance and employing MST as an estimate for the routing capacitance, the power aware clustering problem can be defined as follows. Problem Definition 1*:* For $S = \{s1, \cdots, sn\}$, a set of sequential, create a set of clusters Sclusters = $\{c1, \cdots, cm\}$ that minimizes the following cost function: $_ci \in S$clusters $(\alpha + \text{MST}(ci))$ (2) where α is a user-defined parameter that captures the fixed buffer cost and MST(ci) is the estimate of interconnect capacitance based on MST length of the clock routing. There are constraints due to the available buffer sizes from library $\_sj \in ci\ Csj + \text{MST}(ci) \leq C\text{max}$ (3) where Csj is the gate capacitance of the sequential sj and the Cmax is a user-specified parameter. Choosing sufficiently small Cmax ensures that the subsequent sizing step can meet the target delay, slope, and skew constraints for the clock. Additional constraints that help meeting the skew requirements in the routing step can be applied as follows: $|xsi - xsj| + |ysi - ysj| \leq M\text{max}$, for $si, sj \in ci$ (4) where $xsi(sj)$, $ysi(sj)$ are the coordinates of

location of the sequential *si* (*sj*) and *M*max is a user-specified constant accounting for process and temperature corners. This skew-control constraint, also known as bounding box constraint, ensures that the wire delays from the output of the buffer to any two sequential in the same cluster are bounded, and therefore, the skew due to the effect of process variations on interconnects is bounded. The constraint may also be specified individually for either *x* or *y*-direction. Instead of this constraint, one can also use wire delay or slope constraint directly. To minimize the cost function subject to above constraints, we propose a greedy clustering algorithm that runs in polynomial time.

Clustering algorithm for low power clock tree synthesis
Input: A set of sequentials S = {s1, · · · , sn} with locations
(Xsi, Ysj) and loads Csi for i = 1, · · · , n
Output: A set of optimal Clock tree Synthesis clusters Scluster = {c1, · · · , cm}
1: Create and initialize a set of clusters, Scluster = {c1, · · · , cn}
such that ci = {si} for i = 1, · · · , n
2: Create graph G(S,E), where E = {e(si, sj) : i, j ∈ {1, · · · , n} and i _= j}
3: Find middle point P: (Xsi, Ysj)
4: Sort the set of edges E in ascending order and find (MinXsi, minYsj)
5: Sort the set of edges E in descending order and find (MaxXsi MaxYsj)
6: for all e(si, sj) € E do
7: MinXsi ← si; MinYsj ← sj; MaxXsi ←si; MaxYsj ←sj;
8: If MinXsi > si then
9: MinXsi ←si
10: end if
11: if MinYsj > sj then
12: MinYsj←sj
13: end if
14: if MaxXsi < si then
15 MaxXsi ← si
16: end if
17: If MaxYsi < sj then
18: MaxYsj ←sj
19:end if
20: end for
21: Xsi ← (MaxXsi – MinXsi) / 2
22:Ysi ←(MaxYsj – MinYsj) / 2
23: Bcluster ←P(Xsi,Ysj)
24:Create Clock tree synthesis in Bcluster, Bcluster = {B1,…. Bn}
25:Scluster = {c1, · · · , cm}.

IV. ARCHITECTURAL DESIGN FLOW

A. To Generate Netlist File and sdc File

The Verilog Benchmark files that are needed before starting the flow are Design constraint file and Timing library files. First, the verilog benchmark file is read. Then we need to elaborate the design to the top module. Now, the constraint file is read. Then, the top module of the constraint file is mapped.

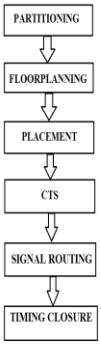

Fig 4.1 VLSI physical design flow

B. Place and Route the Design in Encounter

a) Floor plan Design
The first step in the physical design flow is Floor-planning. Floor-planning is the process of identifying structures that should be placed close together, and allocating space for them in such a manner as to meet the sometimes conflicting goals of available space (cost of the chip), required performance, and the desire to have everything close to everything else. The size height by width ratio to 1.The core utilization is assigned to 0.7 and the core to IO boundary values.
b) Placement Design
Process of finalizing the exact location and orientation of standard cells in design. Standard cells are placed in rows Virtual placement is done for analysis of floor-plan aiming and congestion driven analysis have been done for
Optimization

C. I2C SLAVE CONTROLLER
To achieve the implementation of CTS .There is a need to measure the efficiency of dynamic power with a design which is well known and standardized .Hence a I2c slave sequential circuit is considered for the

experimentation and obtained results are put forth. Simulation, synthesis and physical designing of proposed design has been done using digital cadence. RC compiler has been used for synthesis of design. Physical designing is done using Encounter Digital Implementation System

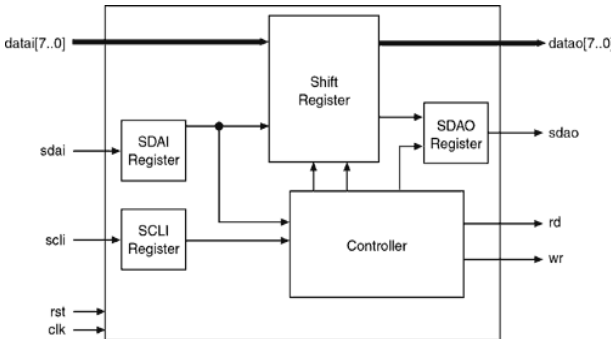

Fig 4.2 Block diagram of I2C slave controller

V. SIMULATION AND OUTPUT

A. FLOOR-PLANNING

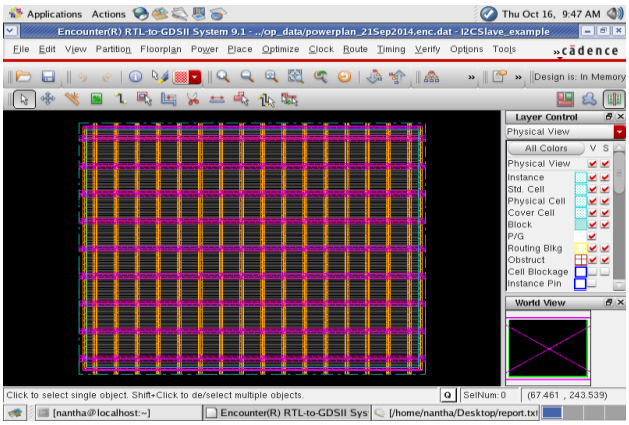

Fig 5.1 Implementation of floor-planning in encounter

B. PLACEMENT

Placement is an essential step in electronic design automation - the portion of the physical design flow that assigns exact locations for various circuit components within the chip's core area. An inferior placement assignment will not only affect the chip's performance but might also make it non manufacturable by producing excessive wire length, which is beyond available routing resources. Consequently, a placer must perform the assignment while optimizing a number of objectives to ensure that a circuit meets its performance demands.

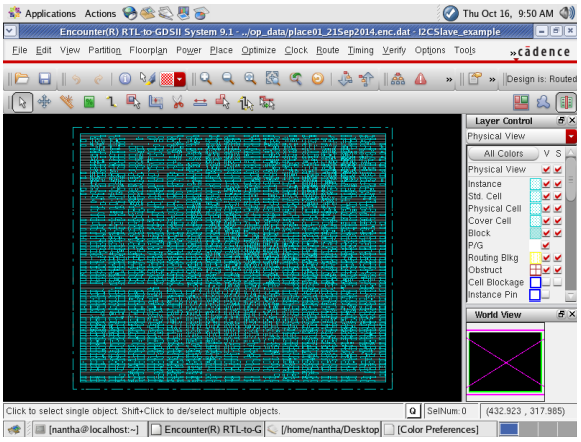

Fig 5.2 Implementation of placement in encounter

C. CLOCK TREE SYNTHESIS

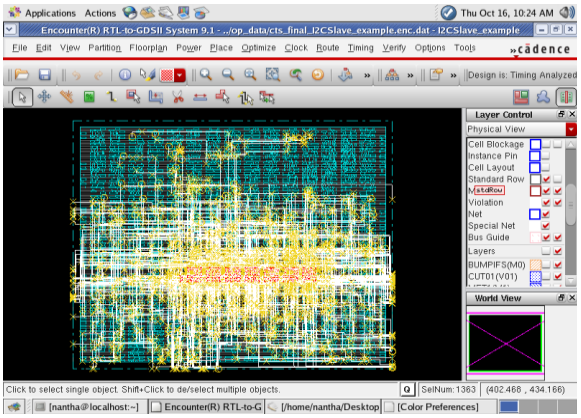

Fig 5.3 Implementation of CTS in encounter

D. CLOCK TREE ANALYSIS

Global skew achieves zero skew between two ynchronous pins without considering logic relationship. Local skew achieves zero skew between two synchronous pins while considering logic relationship. If clock is skewed intentionally to improve setup slack then it is known as useful skew.

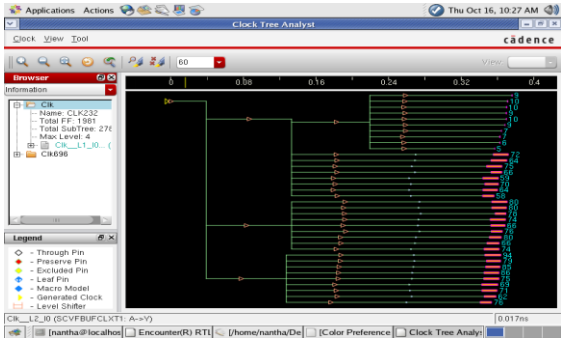

Fig 5.4 clock tree analysis in encounter

E. TIMING ANALYSIS

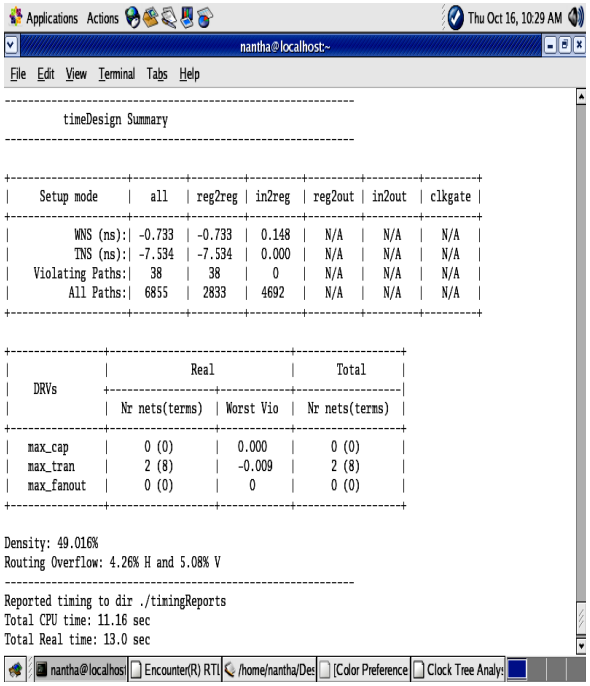

Fig 5.5 Static timing analysis report

F. POWER ANALYSIS

To Minimizing the switching activity reduces the average power dissipation.

Switching Power = 30.29 mW
Internal Power = 28.97 mW
Leakage Power = 0.96 mW
Total Power = 60.23 mW

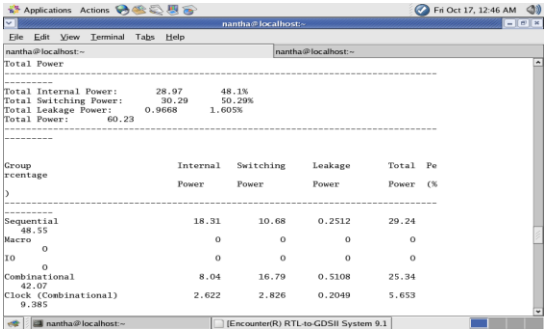

Fig 5.6 Power analysis report

VI. CONCLUSION

Clock tree synthesis is applied in the I2C sequential circuit. The goal of clock tree synthesis (CTS) with min clock skew and min clock latency is implemented. Power and timing analysis have been done using default clustering..The power aware clustering algorithm will be used in the next phase to reduce the overall power consumption with minimum timing constraints. The results will be compared with these results

REFERENCES

[1] N. Kurd, J. Barkatullah, R. Dizon, T. Fletcher, and P. Madland, "A multigigahertz clocking scheme for the pentium 4 microprocessor,"*IEEE J. Solid-State Circuits*, vol. 36, no. 11, pp. 1647–1653, Nov.2001.

[2] R.-S. Tsay, "An exact zero-skew clock routing algorithm," *IEEE Trans.Comput.-Aided Des.*, vol. 12, no. 2, pp. 242–249, Feb. 1993.

[3] M. Edahiro, "A clustering-based optimization algorithm in zero-skew routings," in *Proc. DAC*, Jun. 1993, pp. 612–616.

[4] A. Vittal and M. Marek-Sadowska, "Power optimal buffered clock tree design," in *Proc. DAC*, Jun. 1995, pp. 497–502.

[5] A. Mehta, Y.-P. Chen, N. Menezes, D. Wong, and L. Pileggi, "Clustering and load balancing for buffered clock tree synthesis," in *Proc. ICCD*,Oct. 1997, pp. 217–223.

[6] Y.-L. Chuang, H.-T. Lin, T.-Y. Ho, Y.-W. Chang, and D. Marculescu,"Price: Power reduction by placement and clock-network co-synthesis for pulsed-latch designs," in *Proc. ICCAD*, Nov. 2011, pp. 85–90..

[7] T. Mittal and C.-K. Koh, "Cross link insertion for improving tolerance to variations in clock network synthesis," in *Proc. ISPD*, Mar. 2011, pp. 29–36. driven post-silicon-tunable clock-tree synthesis," in *Proc. ICCAD*, Nov.2005, pp. 575–581.