Journal of Applied Sciences

ISSN 1812-5654

Journal of Applied Sciences 9 (11): 2150-2155, 2009
ISSN 1812-5654
© 2009 Asian Network for Scientific Information

Fast Clock Tree Generation Using Exact Zero Skew Clock Routing Algorithm

[1]M.B.I. Reaz, [2]M.I. Ibrahimy and [1]N. Amin
[1]Department of Electrical, Electronic and Systems Engineering,
Universiti Kebangsaan Malaysia, 43600 UKM, Bangi, Selangor, Malaysia
[2]Department of Electrical and Computer Engineering, International Islamic University Malaysia,
53100 Kuala Lumpur, Malaysia

Abstract: 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. The methodology has been tested using a real design database and resulting in a significant improvement in the through put time required to complete the clock tree generation. This improvement is attributed to the ability to generate clock tree on much smaller portions of clock nets that supports of speeding up the clock tree generation process in IC design.

Key words: Zero skew, clock routing, clock tree generation, IC design

INTRODUCTION

At a deep submicron (DSM) silicon technology levels, it is now feasible to integrate all major function of an end product in a single system-on-a-chip (SOC) silicon die (Rashinkar *et al*., 2001). However, almost all of these chips are still designed based on synchronous clock design. Physically, the clock signal is distributed from an external pad to all similarly clocked synchronizing elements through a distribution network that includes clock distribution logic and interconnects. It serves to unify the design by determining the precise instance in time that the synchronizing elements change state. In order to improve chip performance i.e., to make the chip operates at faster clock speed; the chip designer must ensure that the clock signal is distributed to all the synchronizing elements in the design at the same time. Non-optimal clock behavior is typically caused by either one of these two phenomena: the unbalanced routing to the chip's synchronizing elements, or the non-symmetric behavior of the clock distribution logic (Jackson *et al*., 1990). The problem is further complicated by the massive size of modern chip design and the fact that the design usually utilizes more than one clock signal to sync up its function. Each of these clocks may operate at different frequencies (Chi and Huang, 2000). Therefore, the common challenge faced by all chip designers is how can they create a perfect clock distribution scheme that could guarantee all the timing elements scattered all over the chip are activated by the same edge of the clock signal simultaneously. The maximum difference between the arrival times from the clock source to the elements is defined as the clock skew (Chao *et al*., 1992; Chen and Wong, 1996; Wei *et al*., 2006). The ideal case is to have all the elements sync up at the same time, which would mean no skew.

The benefits of a zero skew clock distribution are well known and have been proven to affect the performance of a chip significantly. Many researches have pointed out that minimizing clock skew is the most important task when designing a clock network. Jackson *et al.* (1990), Kahng *et al.* (1991) and Tsay (1991) have led to many other studies and researches on how to balance a clock network and achieve the zero clock skew targets. More recent studies, such as Pullela *et al.* (1995), Chen *et al.* (1999) and Chakraborty *et al.* (2006) also reveal that reducing clock skew can help to minimize power consumption of large chips, which is very important for chips that are designed for low power application (longer battery life) such as mobile computers and cell phones.

Corresponding Author: Mamun Bin Ibne Reaz, Department of Electrical, Electronic and Systems Engineering, Universiti Kebangsaan Malaysia, 43600 UKM, Bangi, Selangor, Malaysia
Tel: +603-83216309 Fax: +603-83216146

As the chip size grows (in term of increasing number of functional elements), the clock nets get bigger as well. Clock nets have bigger fan-out and have to be distributed over larger areas. It has been observed that existing Clock Tree Generation (CTG) tool performance has deteriorated as the clock net fan out size grow bigger and bigger. The tool execution time is getting slower and slower and it is taking more iteration before the user can get a result that matches the desired outcome. It is suspected that the tool may have hit the limit of its computation capability and is not able to handle nets with large fan-out. Although the CTG tool has been used extensively and successfully before, this capability limitation is slowing the design process.

Since, the problem is the CTG tool capacity, the logical solution is to break the clock nets into smaller parts. This can be accomplished by partitioning the chip into several pseudo-partitions at the layout level, based on the cells placement, as opposed to partitioning at the RTL level, which will be a real design partitioning effort. Partitioning at RTL level will involve some changes to the chip architecture and would increase the complexity of this solution.

The pseudo-partitioning task will be executed after the completion of cells placement stage. The design engineer can output the placement information of the chip and analyze and decide the best way to partition each clock net. The engineer must also make sure that the size of each pseudo-partition should be well below the upper limit of what the CTG tool is capable of handling.

For each of the pseudo-partition, a new buffer will be inserted to act as a new clock source point for each pseudo-partition. Each pseudo-partition will have its own clock source point and that clock point will drive a smaller number of fan-out. The location of the clock source point will be greatly influenced by the floor plan of the chip and the placement location of the fan-out cells as well. Since, the clock net is smaller, the CTG tool should perform well within expectation.

There are many clock routing algorithms that have been proposed throughout the years. The aim of all of these algorithms is to deliver a zero or minimal skew clock routing and distribution.

One of the earliest clock routing algorithms is the H-tree clock routing algorithm. While the H-Tree clock routing algorithm is widely used in the IC industry and their application is not suitable for IC design because cells placement in IC design is not symmetrical (Bakoglu *et al*., 1986). This fact is also supported by the power PC microprocessor design team. The team has conducted experiment on their chip and concluded that clock tree generation methodology with balance router ensure quick turnaround and produce more than 40% improvement over H-Trees (Carrig *et al*., 1997; Metra *et al*., 2003).

The Method of Means and Median (MMM) algorithm by Jackson *et al*. (1990) has been proposed to overcome the H-Tree algorithm shortcoming but it also has its weaknesses. The difference in wire length from the clock source to each leaf cell can be as large as half the diameter of the chip. The algorithm effectiveness depends heavily on how the cut direction is decided, thus adding lethargy to the algorithm implementation.

The Recursive Geometric Matching (RGM) algorithm was proposed in 1991 by Kahng *et al*. (1991). It has been proposed to fix the unbalanced wire length problem seen in the MMM algorithm. This algorithm always yields clock tree with perfectly balance path length for trees of two, three, or four terminals, but does not necessarily minimize the skew as it did not consider the cell loading. Like the MMM algorithm, the algorithm cannot guarantee a balanced wire length on certain cases and requires work around that reduces its effectiveness.

The Exact Zero Skew (EZS) algorithm was first proposed by Tsay (1991), from IBM's (International Business Machine) T.J. Watson Research Center. It considers delay balance instead of the wire length balance has been proposed to deliver better skew performance. The EZS algorithm adopts a bottom up process similar to that of the RGM algorithm. This algorithm achieves delay balance by elongating wires that have smaller delays, therefore, does not require any look ahead techniques that can slow down the execution.

In order to have a fair comparison, simulation exercise to test out the three different algorithms, namely, the MMM, the RGM and the EZS algorithm, to route a clock net with 8 leaf cells has been performed. The wire used for routing was 1 micron wide and the wire resistance per unit length was 1 mΩ and wire capacitance per unit length was 0.075 fF. Each unit length was equivalent to 1 μm. The die area was 8000 by 8000 μ. The overall comparison is shown in the Table 1.

The simulation results clearly show that the EZS algorithm gives the smallest skew (0.004233 psec) and its amount is negligible. However, in this test case, the maximum path delay and the amount of wire length used to route the nets using EZS algorithm are only slightly lower compared to the other algorithm (3.7% difference in maximum path delay difference and 0.4% in wire length). This is consistent with what is reported by Tsay (1991). Based on all these considerations, the EZS algorithm is

Table 1: Summary of the comparison

Algorithm	Max path delay (psec)	Skew (psec)	Total wire length (μm)
MMM	16.78469	1.883	88000
RGM	16.21875	0.285	88000
EZS	15.61857	0.004233	87610

recommended and proposed to be deployed in the implementation of the zero skew clock net routing tools.

MATERIALS AND METHODS

Implementation of the solution: The proposed solution is called Zero Skew Clock Routing (ZSCR) methodology. The methodology is presented in the flowchart shown in Fig. 1. It can actually be categorized in three stages. The first stage is the pseudo-partitioning stage. In this stage, the user analyzes the placement results and identifies how to pseudo-partition the clock networks. The user has to also determine the location of the new clock source for each pseudo-partition. The second stage is the clock trunk routing stage. For this stage, a software program based on the EZS algorithm had been created to perform the zero skew routing between the original clock sources to the new pseudo-partition clock sources. This part of the net is referred to as the clock trunk as it now became the main branch of the clock net. The third stage is the database update stage. In this stage, the results from stage 1 and stage 2 are integrated back into the database. This methodology is implemented in three stages.

The pseudo-partitioning stage: In the first stage, a couple of Perl scripts were written to help the user extract and analyze placement data as well as determine the pseudo-partition boundary and the location of the new clock source. A set of guidance on the decision making process of determining the pseudo-partition boundary and clock buffers location was given as well.

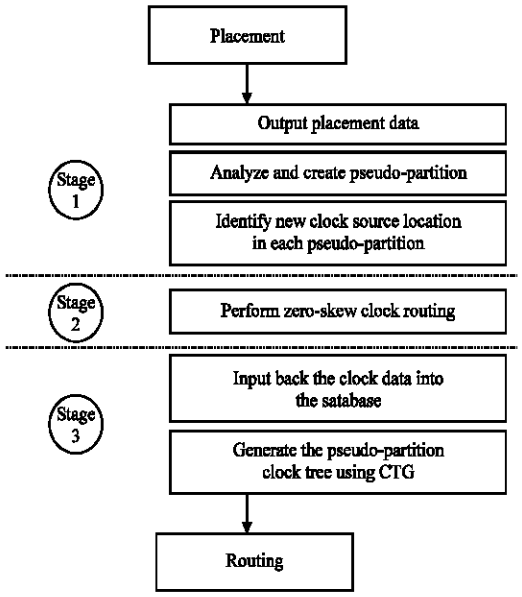
Placement
Output placement data
Stage 1
Analyze and create pseudo-partition
Identify new clock source location in each pseudo-partition
Stage 2
Perform zero-skew clock routing
Stage 3
Input back the clock data into the satabase
Generate the pseudo-partition clock tree using CTG
Routing

Fig. 1: Zero skew clock routing flowchart

The clock trunk routing stage: For this stage, a software tool called Mojiii was created to perform the zero skew routing between the original clock sources to the new pseudo-partition clock sources. The routing algorithm used in this tool is based on the EZS algorithm that was authored by Tsay (1991) from IBM's (International Business Machine) T.J. Watson Research Center. The tool is written using Microsoft Visual Basic. The main input to this tool is the location of the original and pseudo-partition clock sources and the input pin capacitance of the pseudo-partition clock source. One of the outputs from the tool is a command file that instructs how to implement the routing in automatic place and route-graphical user interface (APR-GUI).

Updating database: In the third and final stage, another set of Perl scripts was coded to integrate back the results from stage 1 and stage 2 back into the database. After completing all the three stages, the user continues with the normal clock tree generation process. Since, the clock nets have been partitioned into smaller portions, the clock tree generation process completes faster and requires less iteration to meet the skew and delay specification.

RESULTS AND DISCUSSION

Results of calibration test cases: To calibrate the tool accuracy and validate the legality of the methodology and its results, a couple of test cases have been conducted. The test cases use the g Clk 400 m and sR32k nets. The nets are chosen because of their placement distribution reflect typical clock distribution on the chip. Net g Clk400 m has 2231 fan-out or leaf cells and net sR32k has 687 leaf cells. The number of leaf cells in this two test case is not very high, to allow faster evaluation, less tedious and more accurate result validation. The clock frequency for gClk400 m is 400 MHz, the clock period is 2.5 nsec and the maximum skews allowed are set to 5% of the clock period, which are 125 psec in this case. For the sR32k net, the clock frequency is 32 MHz, the clock period is 31.25 nsec and the maximum skew allowed is also set to 5% of the clock period, which are 1562.5 psec in this case.

The net is routed in Mojiii to ensure zero skew routing from the clock source to the new buffers. The resistance per square length value is set to 0.001 mΩ μm^{-1} and the capacitance per length is set to 0.075 fF μm^{-1}. These numbers are derived from the resistance per square value and the capacitance per square value of the routing layer, assuming that all the routing will be done with 1 μm wire. The results of the routing are shown in the Fig. 2 for gClk400 m net and in the Fig. 3 for sR32k net.

From the results calculated in Mojiii, the path delay for net gClk400 m, from the clock source to its leaf cells is 4.283 psec, with zero (negligible) skew and for net sR32k from the clock source to its leaf cells is 6.560 psec, with zero (negligible) skew, as expected. The routing for both nets is then implemented on the real database using two different routing layers. The results are shown in the Table 2 and 3.

From the extracted results, it is observed that the output to input pin path delay is about 10 to 15% greater than what is estimated using the Lumped Delay model in the EZS algorithm. The skew is not exactly zero, but rather some very small value, less than 0.01% of the clock period. Nevertheless, the skew and path delay are acceptable and would not impair the implementation of the proposed solutions.

Another item that needs to be evaluated is the total time needed to generate the clock tree using this new methodology. The results are shown in Table 4. The throughput time is about 13 to 18% better for these test cases. However, it is believed that it will affect more significantly on larger clock nets.

Results of evaluation on a chip database: The ZSCR methodology was deployed on one of the chips. The chip was fabricated in 0.35 μ (L_{eff} = 0.25 microns), 2.5 V CMOS technology with 6 metal layers. The chip die size was 7.2×5.5 mm and contained 2.8 million transistors. The clock distribution network has to serve a large number of macro cells (18) along with approximately 32,000 master slave latch-pairs. The frequency of the chip's main clock is 250 MHz.

The main clock was first balanced using the original CTG tool. Then the balancing exercise was repeated using the ZSCR Methodology. The results from both run are compared in Table 5.

Based on the results, the maximum path delays and maximum skews show some improvement, but are not significantly different. This is expected as the main underlying tools for balancing the clock net is still the same, which is the CTG tool. However, there is a significant 50% improvement in the through put time. This improvement is attributed to the ability to run CTG on much smaller portions of clock nets and to the ability to

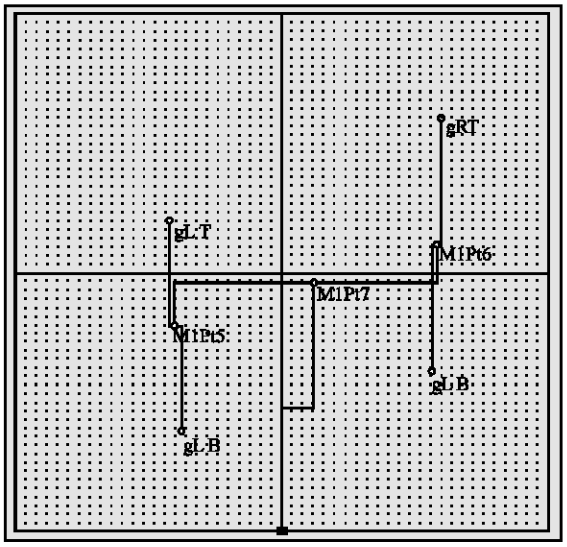

Fig. 2: Routing results for net gClk400 m

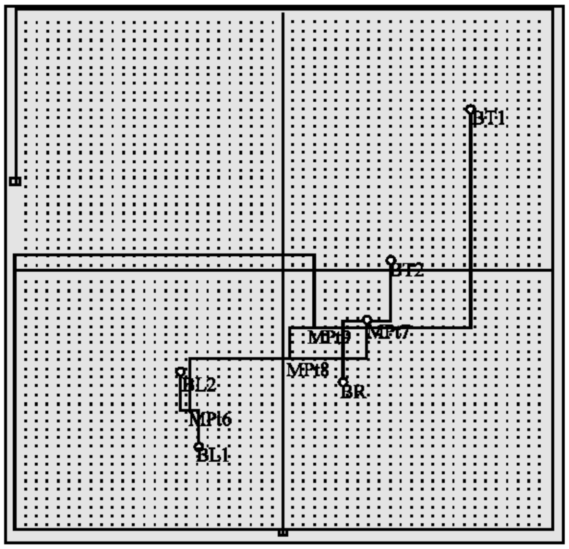

Fig. 3: Routing results for net sR32k

Table 2: Extracted results for net gClk400 m

Cell	X	Y	Input capacitance (fF)	Output to input pin path delay (psec)
gClk400 m	2504	3	-	-
gLB	1545	1150	70	4.928844
gLT	1445	3600	70	4.950264
gRB	3910	1855	70	4.841353
gRT	3998	4795	70	4.862765
Longest delay				4.950264
Shortest delay				4.841353
Maximum skew				0.108911
Clock period (400 MHz)				2500
Percentage of skew over clock period				0.00436%

Table 3: Extracted results for net sR32k

Cell	X	Y	Input capacitance (fF)	Output to input pin path delay (psec)
sR32k	5	4006	-	-
BL1	1720	950	70	7.437194
BL2	1550	1800	70	7.474003
BR	3050	1700	70	7.371831
ST2	3500	3100	70	7.338573
ST1	4250	4850	70	7.304272
Longest delay				7.470272
Shortest delay				7.304272
Maximum skew				0.1657316
Clock period (32 MHz)				31250
Percentage of skew over clock period				0.0005303%

Table 4: Throughput-time comparison

Net	Original (min)	Mojiii (min)	Improvement (%)
gClk400 m	261	226	13.40
sR32k	145	119	17.93

Table 5: Results comparison within both run

Flow	Maximum path delays (nsec)	Maximum skew (psec)	Throughput time (days)
Original	12.5621	180	14
Zero skew	12.2873	174	7

run it concurrently, i.e., running five smaller nets on five workstations in parallel.

The results of this evaluation validate that the ZSCR methodology has been correctly implemented and has produced valid results. It also confirms that the deployment of the methodology has achieved its goal of speeding up the clock tree generation process in IC design.

CONCLUSION

The ZSCR methodology has been shown to have helped the design team improve their overall throughput time when balancing the main clock network on the chip. There are still many areas of improvement that can be implemented.

The pseudo-partitioning stage can be further automated to eliminate the need for the user to manually analyze the placement. The efficiency of stage two of the methodology can be further improved to provide more accurate delay estimations and address other realistic layout concerns. One of the layout concerns is that the EZS algorithm does not strive for wire length minimization. Another area of improvement that should be implemented is wire-sizing.

In short, there are many potential improvements that can be implemented to make this methodology more effective. Clock network distribution will remain very important in chip design, therefore, efforts poured in making it more efficient are efforts well spent.

REFERENCES

Bakoglu, H., J.T. Walker and J.D. Meindl, 1986. A symmetric clock distribution tree and optimized high speed interconnections for reduced clock skew in ULSI and WSI circuit. Proceedings of the IEEE International Conference on Computer Design, Oct. 5-10, New Jersey, USA., pp: 118-122.

Carrig, K.M., A.M. Chu, F.D. Ferraiolo, J.G. Perovick, P.A. Scott and R.J. Weiss, 1997. A clock methodology for high-performance microprocessors. Proceedings of the IEEE Custom Integrated Circuits Conference, May 5-8, Santa Clara, CA, USA., pp: 119-122.

Chakraborty, A., P. Sithambaram, K. Duraisami, A. Macii, E. Macii and M. Poncino, 2006. Thermal resilient bounded-skew clock tree optimization methodology. Proceedings of the Conference on Design, Automation and Test in Europe, Mar. 6-10, Munich, Germany, pp: 832-837.

Chao, T.H., Y.C. Hsu and J.M. Ho, 1992. Zero skew clock routing with minimum wirelength. IEEE Trans. Circuits Syst., 39: 799-814.

Chen, R.Y., N. Vijaykrishnan and M.J. Irwin, 1999. Clock power issues in system-on-a-chip designs. Proceedings of the IEEE Computer Society Workshop on VLSI '99, Apr. 8-9, Orlando, FL, USA., pp: 48-53.

Chen, Y.P. and D.F. Wong, 1996. An algorithm for zero skew clock tree routing with buffer insertion. Proceedings of the European Design and Test Conference, Mar. 11-14, Paris, France, pp: 230-236.

Chi, M.C. and S.H. Huang, 2000. A reliable clock tree design methodology for ASIC designs. Proceedings of IEEE 1st International Symposium on Quality Electronic Design, Mar. 20-22, San Jose, CA, USA., pp: 269-274.

Jackson, M.A.B., A. Srinivasan and E.S. Kuh, 1990. Clock routing for high-performance ICS. Proceedings of the 27th ACM/IEEE Design Automation Conference, Jun. 24-28, Orlando, Florida, USA., pp: 573-579.

Kahng, A., J. Cong and G. Robins, 1991. High performance clock routing based on recursive geometric matching. Proceeding of the 28th ACM/IEEE Design Automation Conference, Jun. 17-21, San Francisco, California, USA., pp: 322-327.

Metra, C., T.M. Mak and D. Rossi, 2003. Clock calibration faults and their impact on quality of high performance microprocessors. Proceedings of the 18th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, Nov. 3-5, Boston, MA, USA., pp: 63-70.

Pullela, S., N. Menezes and L.T. Pillage, 1995. Low power IC clock tree design. Proceedings of the IEEE Custom Integrated Circuits Conference, May 1-4, Santa Clara, CA, USA., pp: 263-266.

Rashinkar, P., P. Paterson and L. Singh, 2001. System-on-a-Chip Verification: Methodology and Techniques. 4th Edn., Kluwer Academic Publishers, New York, ISBN: 0-7923-7279-4.

Tsay, R.S., 1991. Exact zero skew. Proceedings of the IEEE International Conference on Computer Aided Design, Nov. 11-14, Santa Clara, CA, USA., pp: 336-339.

Wei, X., Y. Cai and X. Hong, 2006. Clock skew scheduling under process variations. Proceedings of the 7th International Symposium on Quality Electronic Design, Mar. 27-29, San Jose, CA, USA., pp: 237-242.