

# Flat Multi-Million Gate ASIC Designs Using ASTRO

Kazutaka Murakami, Hidetaka Minami, Zenji Oka  
CAD Engineering Section, Imaging System LSI Development Center  
Electronics Devices Company  
RICOH COMPANY, LTD.  
{kazutaka.murakami, hidetaka.minami, zenji.oka}@nts.ricoh.co.jp

## ABSTRACT

Multi-million gate ASICs are typically designed by hierarchical layout style because of the huge memory usage and long CPU time required in flat layout style. However, it is too complicated and takes longer design time, especially for severe timing closure. In this paper, the results of multi-million gate ASICs (up to 5million) designed by APOLLO/SATURN and ASTRO in flat layout style are shown respectively. Also the limitation of APOLLO/SATURN and ASTRO's higher capability are discussed.

## 1. INTRODUCTION

Due to sophisticated system requirements, the circuit scale of system LSI is rapidly increasing and multi-million gate ASICs are becoming the mainstream. For designing such large ASICs, most ASIC designers are trying to use the hierarchical layout method. This is because if we use the flat layout method, it takes much memory and long CPU time, and it becomes almost impossible to achieve timing convergence [1]. In the hierarchical layout flow, multi-million gate circuits are partitioned into several cores sized half to one million gates so as to be acceptable for implementation on current EDA environments [2]. Partitioning also enables us to get accurate estimation of wire length, and timing convergence can be done with a few iterations of synthesis [3][4]. However, it makes the design flow too complicated to achieve QTAT of ASIC design. Furthermore, some issues such as decision of core pin assignment and CTS across the core still remain unsolved.

On the other hand, recent innovations of EDA environments have made the capability of flat layout style spread to multi-million gate ASIC's design. ASTRO is Avant!'s most advanced physical design system. In this paper, we present the flat layout results of multi-million gate ASICs by ASTRO to achieve QTAT of ASIC design. Four test cases of 1, 2.5, 3.5, 5 million gate ASICs have been tried in flat layout style using both ASTRO and APOLLO respectively, and the

layout results using APOLLO/SATURN and ASTRO will be shown. Then, the higher capability of ASTRO and the limitation of the flat layout design using APOLLO will follow. Finally, some enhancement requests for ASTRO will be discussed.

## 2. WHAT IS ASTRO ?

ASTRO is Avant!'s most advanced physical design system. According to the catalog specification, ASTRO has the following features.

Design Convergence - ASTRO's new Milkyway[tm]DUO[tm] (Dynamic Unified Optimization) architecture provides an extremely tight integration between the layout (place and route), analysis (timing, noise, rail), and synthesis optimization engines - the key to achieving consistent and predictable design convergence.

UDSM Effects - The advanced PhySiSys[tm] (Physically Accurate, Signal Integrity and Synthesis Optimization) technology eliminates iterations by consistently and continuously accounting for all UDSM effects throughout the design process.

Superior Results - ASTRO provides performance superior to the current market leader, APOLLO-II and SATURN[tm], with an average of 10% faster clock speeds, up to 50% less memory usage, and up to 3X faster design completion time.

High Productivity - Concurrent placement and clock tree synthesis provides the lowest possible clock skew and early timing and congestion impact prediction, a key contributor for productivity improvement.

Simple Migration - Based on the Milkyway common database, ASTRO eliminates migration issues and provides direct "plug-and-play" into Avant!'s SinglePass-SoC[tm] flow and tools.

This paper will certify that these features work.

### 3. FLAT ASIC DESIGN FLOW USING APOLLO/SATURN AND ASTRO

Flat ASIC design flow by APOLLO/SATURN and ASTRO are described in this chapter. The typical ASIC design flow is shown in Figure3-1. It consists of eight steps.

(Step1) At first, designers describe RTL code according to the ASIC specification. After finishing RTL coding , the RTL is verified by NC-Verilog.

(step2) The RTL lint checker is used. It checks the design rules based on RMM (Reuse Methodology Manual). There are several RTL lint checkers in the market, but we use our own because it can be easily customized. It checks DFT (Design for Testability) rules, synthesis coding style, and so on.

(step3,4) JTAG and Memory-BIST circuits are inserted.

(step5) Logic synthesis is performed. Design Compiler or BuildGates is used for synthesis. After Synthesis, PrimeTime is used for STA (Static Timing Analysis).

(step6) Scan and Logic-BIST are inserted.

(step7) Timing-Driven-Layout is executed using APOLLO/SATURN or ASTRO. The detailed flow is shown in Figure 3-2 and Figure 3-3  
(step8) Finally PrimeTime is used for sign off.

Figure 3-2 shows the flat layout flow by APOLLO/SATURN and Figure 3-3 shows the flat layout flow by ASTRO. The biggest difference is High Fanout Nets' (HFN) handling as shown in Figure 3-4. ASTRO has moved HFN synthesis into the PrePlace stage.

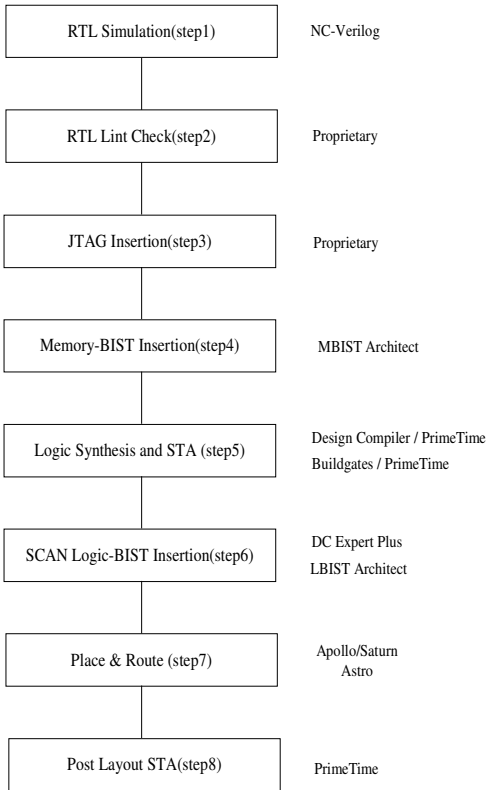


Figure3-1 ASIC design Flow

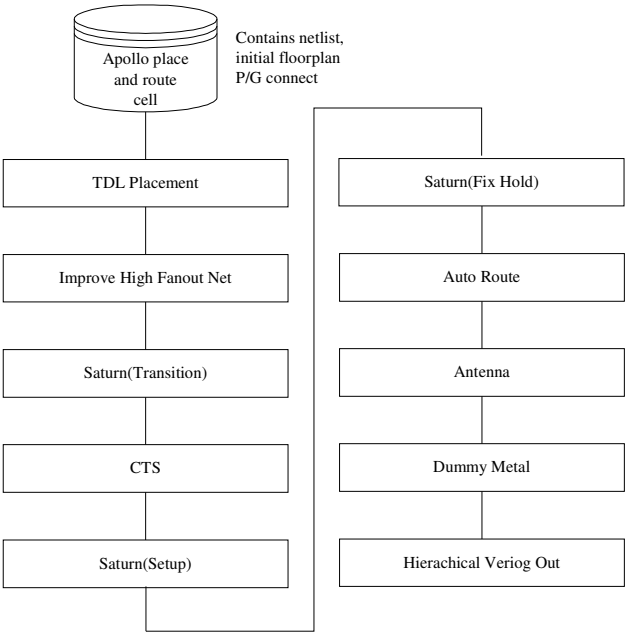


Figure 3-2 APOLLO/SATURN design flow

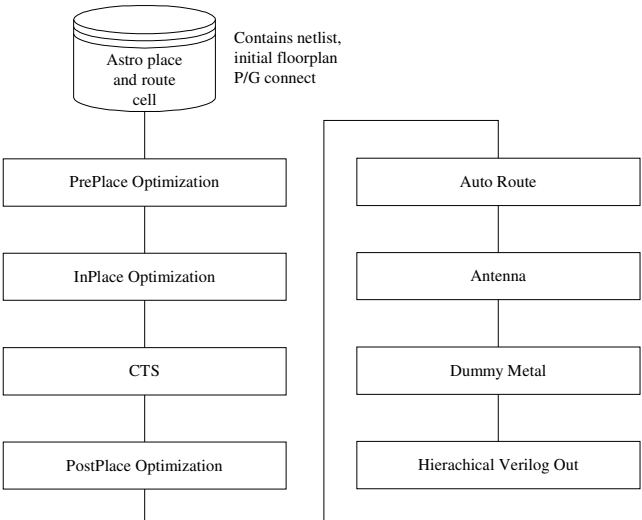
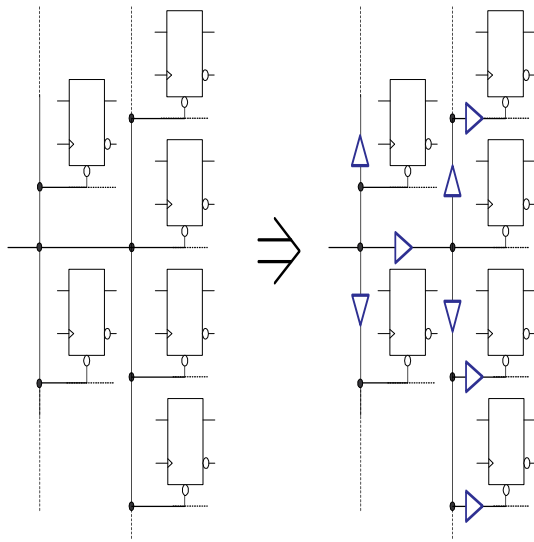


Figure 3-3 ASTRO design flow



Apollo: Placement -> HFNS (Saturn)  
Astro : HFNS(Pre-Place) -> Placement (In-Place)

Figure 3-4 Difference of HFNS between APOLLO/SATURN and ASTRO

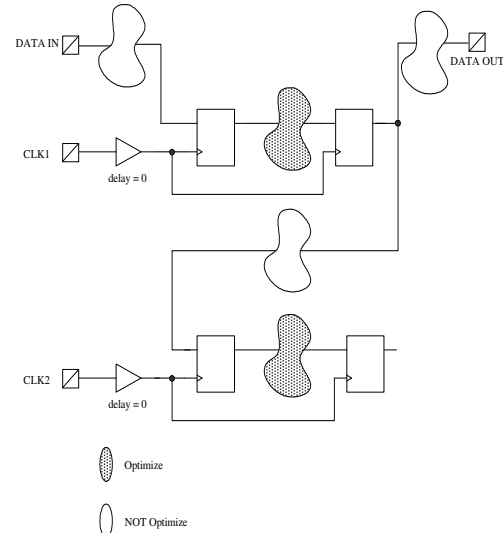


Figure 3-5 In-Place Optimization

APOLLO/SATURN fixes transition delays after real placement. But if High Fanout Nets remain at placement, it takes a long CPU time. So when using APOLLO/SATURN, parasitic RC on HFN must not be extracted by providing virtual delay time using command “tdfSetNetCapTransitionAndDelayTime“. After placement, HFNS are partitioned by CTS. On the other hand, ASTRO executes HFN synthesis automatically at the Pre-Place stage before real placement. Note that if there are clocks that are not declared as clocks, such as JTAG clocks, the load of the clock is shared without considering clock skew. So “don’t touch net constraints” must be added. After Pre-Place, congestion of the cells added by HFN is considered. As mentioned above, Pre-Place is run only for improvement of transition. In-Place performs real placement. But this flow has a problem. When CTS is performed, timing closure is targeted at only for the shaded areas (same clock source) shown in Figure3-5. In this case, clock delay must be 0. Then the timing closure of shaded areas shown in Figure 3-6 (different clock source and AC timing) is improved by Post-Place. In this case, clock propagation delay after CTS as shown in Figure 3-6 must be considered.

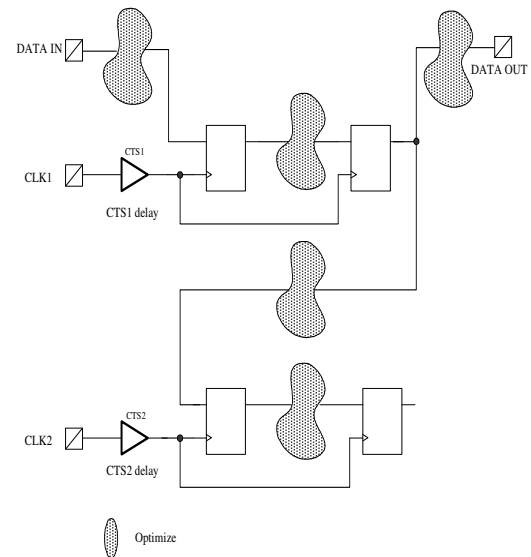


Figure 3-6 Post-Place Optimization

ASTRO’s concurrent placement and CTS are shown in catalog specification, but not implemented yet now. As mentioned above, current ASTRO’s CTS is performed just only after In-Place. Placement considering constraints between different clocks and AC is executed only at Post-Place stage. Just only Post-Place optimization will not be enough for future ASIC with higher density and faster speed. Placement before CTS causes another congestion problem. Before CTS , we use “Search and Refine” command heavily to improve congestion. But After CTS, we cannot use it because the clock skew may be worse than the skew before doing “Search and Refine”. In order to solve these problems, concurrent placement and CTS is necessary. At last, Figure 3-7 shows the concurrent placement and CTS flow. We hope for early release of this capability.

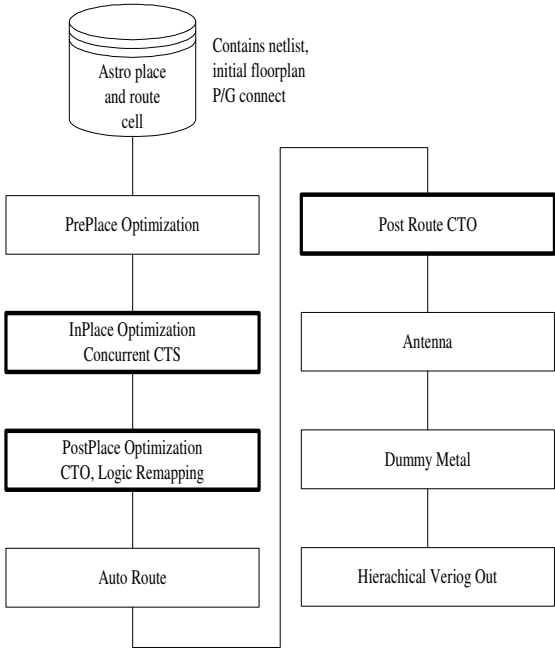
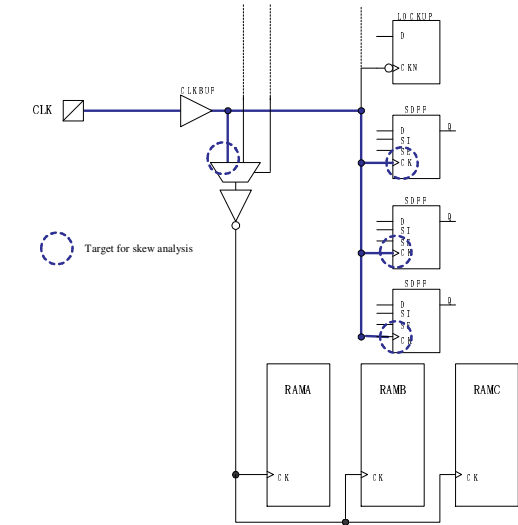
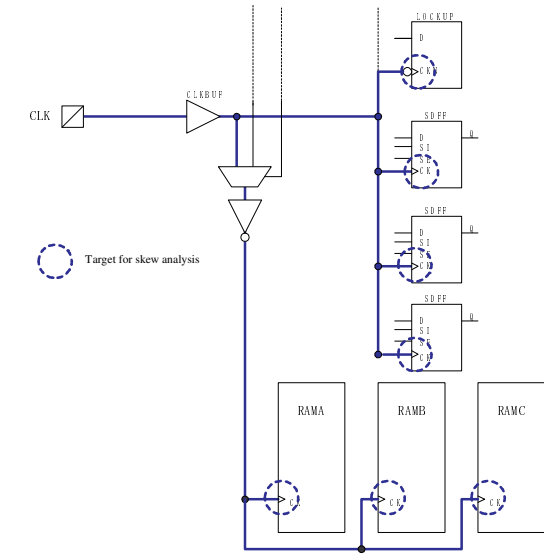


Figure 3-7 Concurrent Placement and CTS flow

Finally, we discuss the difference of clock skew analysis between APOLLO and ASTRO. Figure 3-8 shows the target of skew analysis by APOLLO and ASTRO. The target of skew analysis by ASTRO is only Sync-Pin. And ASTRO has no distinction between the rising edge and the falling edge. To avoid worse clock skew, clock of megacell such as RAMs placed away from standard cell logic should not be included in CTS in ASTRO.



(A) APOLLO



(B) ASTRO

Figure 3-8 Target for skew analysis

#### 4. COMPARISON OF 4 TEST CASES' LAYOUT RESULT BETWEEN APOLLO/SATURN AND ASTRO

In this chapter, comparison of 4 test cases between APOLLO/SATURN and ASTRO is shown. The machine used for the benchmark is SUN Enterprise4500 workstation (CPU 400MHz, Main memory 24GB). APOLLO/SATURN's timing constraints are provided by Synopsys Timing Requirements File. For ASTRO, timing constraints are provided by SDC (Synopsys Design Constraints). In all experiments we have used UMC0.18  $\mu$  m Artisan cell library.

Test Case 1 1 million gates :

The ASIC had been taped out using APOLLO. It is the most popular gate size by APOLLO. It contains 11 clock sources, 60,000 Flip-Flops, max high fanout nets 60,000, max clock frequency 125MHz. ASTRO's run time is 30% faster than APOLLO.

Test Case 2 2.5 million gates :

The ASIC had been taped out using APOLLO. This gate size may be the maximum which APOLLO/SATURN can handle by flat layout style. It contains 8 clock sources, 150,000 Flip-Flops, max high fanout nets 150,000, max clock frequency 256MHz. Current ASTRO has some problems regarding big CTS, so we must divide into 3 parts. We hope this problem will be fixed by now.

Test Case 3 3.5 million gates :

The ASIC is now being designed using ASTRO. It contains 10 clock sources, 200,000 Flip-Flops, max high fanout nets 200,000, max clock frequency 125MHz. APOLLO/SATURN cannot achieve timing convergence. Final negative slack is -0.38ns after running 4 weeks. ASTRO can achieve timing convergence. This ASIC will be taped out by AURORA and may be the first ASIC taped out in the world by ASTRO.

Test Case 4 5 million gates :

This is a test case only for the benchmark, not an actual design. 2clock sources, 70,000 Flip-Flop, max high fanout nets 70,000, clock frequency 25MHz. APOLLO/SATURN cannot achieve timing convergence, and final negative slack is -7.6ns after SATURN is running for 120 hours. ASTRO can achieve timing convergence.

The results of 4 test cases are shown in Table4-1. Figure 4-1 shows the total placement time by both APOLLO/SATURN and ASTRO. Figure 4-1 shows ASTRO's run time increase is almost linear. APOLLO/SATURN's run time is tremendous if gate size becomes over 3million, and does not achieve timing convergence.

Test case4 was also tried using an HP machine. The machine specification is HP J6700 (750MHz), OS HP-UX B.11.00, main memory 16G. The benchmark result is shown in Figure 4-2. This shows that 5million gate ASIC can be designed using a flat layout style within 2 days. This shows that flat layout style up to 5million gate by ASTRO is practical.

Table 4-1 The result of 4 test cases

		Test case 1		Test case 2		Test case 3		Test case 4	
		Apollo	Astro	Apollo	Astro	Apollo	Astro	Apollo	Astro
number of gates	before layout	895,930		2,552,043		3,431,927		5,074,974	
	after layout	971,083	928,598	#####	#####	#####	#####	#####	#####
chip size(mm)		6x6		9.5x9.5		10x10		10x10	
max clock frequency		125MHz		256MHz		125MHz		25MHz	
main clock frequency		33MHz		66MHz		66MHz		25MHz	
Place+CTS (Hr)	Total	18.5	13.25	63	39.5	NA	57.25	NA	120.5
Route (Hr)	Total	8.5	6.9	28	18.5	26.5	18.25	13	13.5

Regarding test case3, we used Logic-BIST. If Logic-BIST is used, there is no limitation of the number of scan chains, and testing time is greatly shortened. But current ASTRO optimizes only one scan chain at a time. So if we divide the scan chain into a great number of scan chains, ASTRO cannot finish routing. This problem will be discussed later in chapter6.

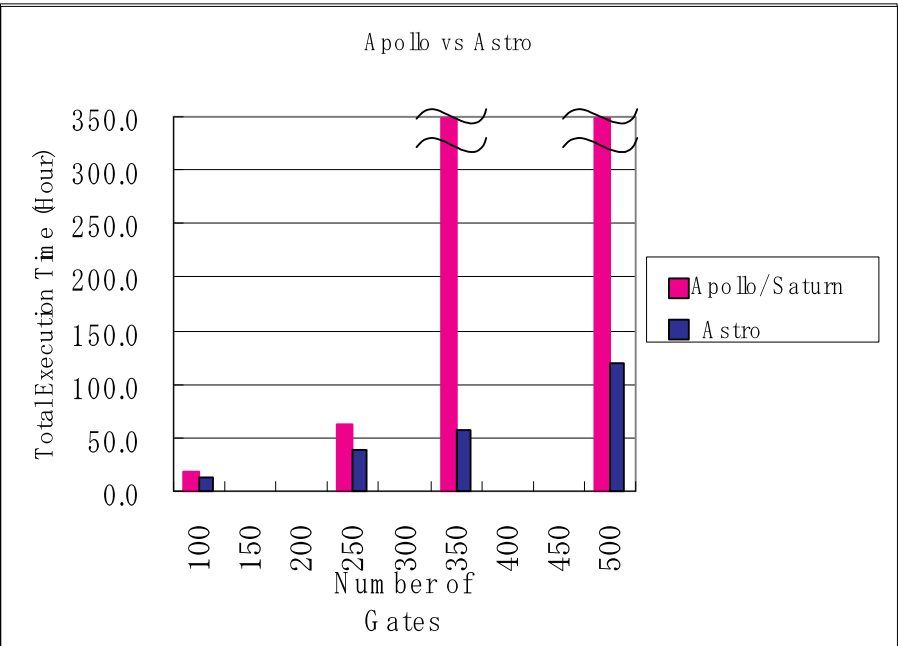


Figure 4-1 Total placement time

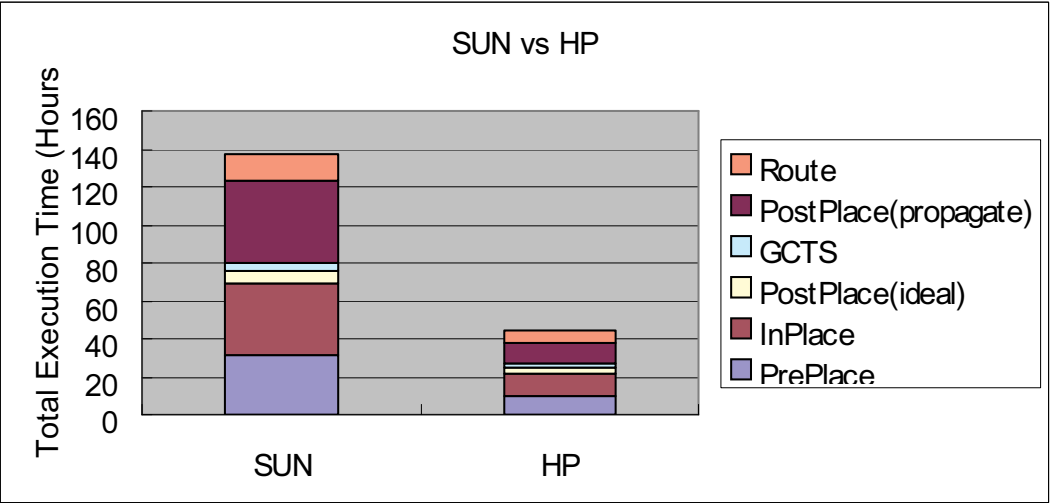


Figure 4-2 Total Execution Time ( Sun vs HP )

## 5. COMPARISON OF DELAY ESTIMATION ACCURACY BETWEEN APOLLO AND ASTRO

Comparison of delay estimation accuracy between APOLLO and ASTRO is discussed in this chapter. 1million gate test case is used for this comparison.

### 5.1 AVERAGE DELAY COMPARISON

First of all, we examine the difference between the delay before and after routing. The generated SDFs(Standard Delay File) by both APOLLO and ASTRO are used for this comparison. Of course both delay calculation conditions before and after routing are exactly the same.

Table 5-1 shows the difference between the delay before and after routing.

Table 5-1 (A) shows the APOLLO's result. Average delays both before and after routing are almost identical. The largest overestimation of delay in all test cases was 4.797 ns , and the largest underestimation was 3.008 ns, and maximum  $\sigma$  (standard deviation) is 0.504 .

Table 5-1 (B) shows the ASTRO's result. Average delay after routing is slower than before by about 2%. This is because lack of technology file tuning . The largest overestimation of delay in all test cases was 1.932 ns , and the largest underestimation was 0.481 ns, and maximum  $\sigma$  (standard deviation) is 0.057 .

The Smaller  $\sigma$  shows ASTRO's delay estimation before routing is more accurate than APOLLO.

### 5.2 CRITICAL PATH DELAY COMPARISON

Table 5-2 shows the critical path slack. Table 5-2 (A) shows the result for APOLLO/SATURN. APOLLO/SATURN cannot achieve complete timing convergence, so we must do another ECO manually. Table 5-2 (B) shows ASTRO's result. Although some timing violations still remain, the result is much better. The result indicates that ASTRO's delay optimization is better than APOLLO.

Figure 5-1 shows a histogram of timing slack. Figure 5-1 (A) shows the APOLLO result. Figure 5-1 (B) shows the ASTRO result. APOLLO has only one stage of placement optimization. ASTRO has three stages of placement optimization, Pre-Place, In-Place and Post-Place. In-Place is performed after load sharing. Apollo performs real placement before load sharing. This seems to be the reason that ASTRO's delay optimization is better than APOLLO.

Table 5-2 critical path slack

	Negative Slack
TDL Place	-3.91
Saturn	-0.60
Route	-0.60
STA	-0.29

(A) APOLLO

	Negative Slack
Pre-Place	-6.64
In-Place	-1.87
Post-Place	-0.31
Route	-0.31
STA	0.00

(B) ASTRO

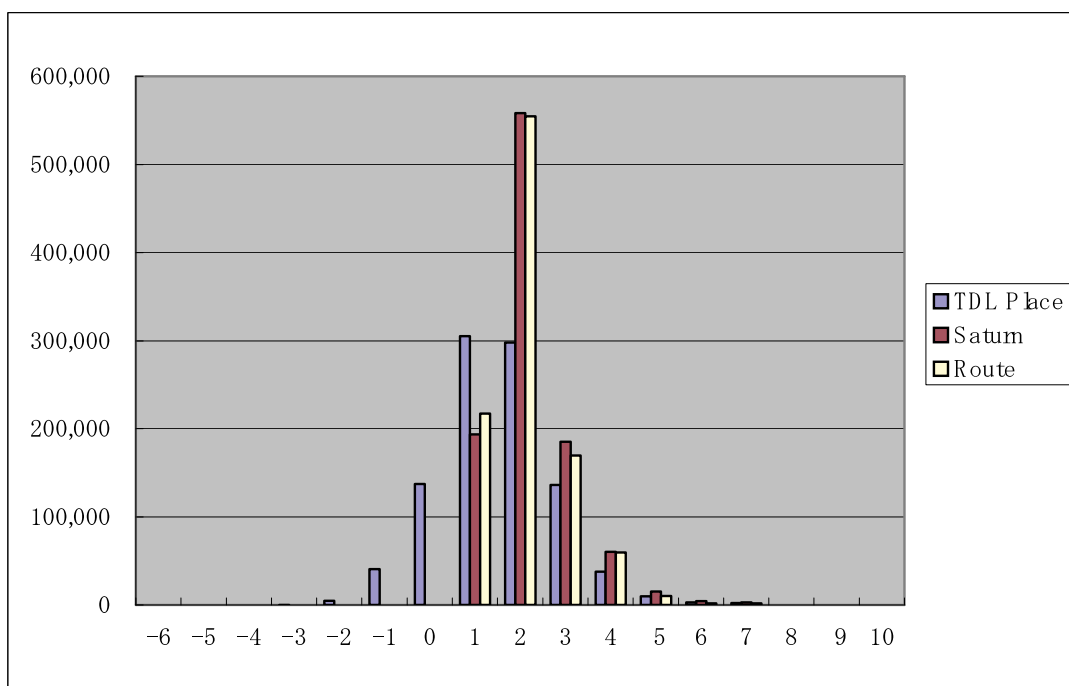
Table 5-1 The difference between the delay before and after routing

speed	Edge	Average(%)	$\sigma$	Largest Delay Overestimation(ns)	Largest Delay Underestimation(ns)
MIN	LH	99.620	0.139	2.019	0.825
	HL	100.240	0.504	1.543	0.827
TYP	LH	99.690	0.020	3.129	1.954
	HL	100.760	0.335	2.410	1.955
MAX	LH	100.380	0.080	4.797	3.008
	HL	100.760	0.080	3.739	3.006

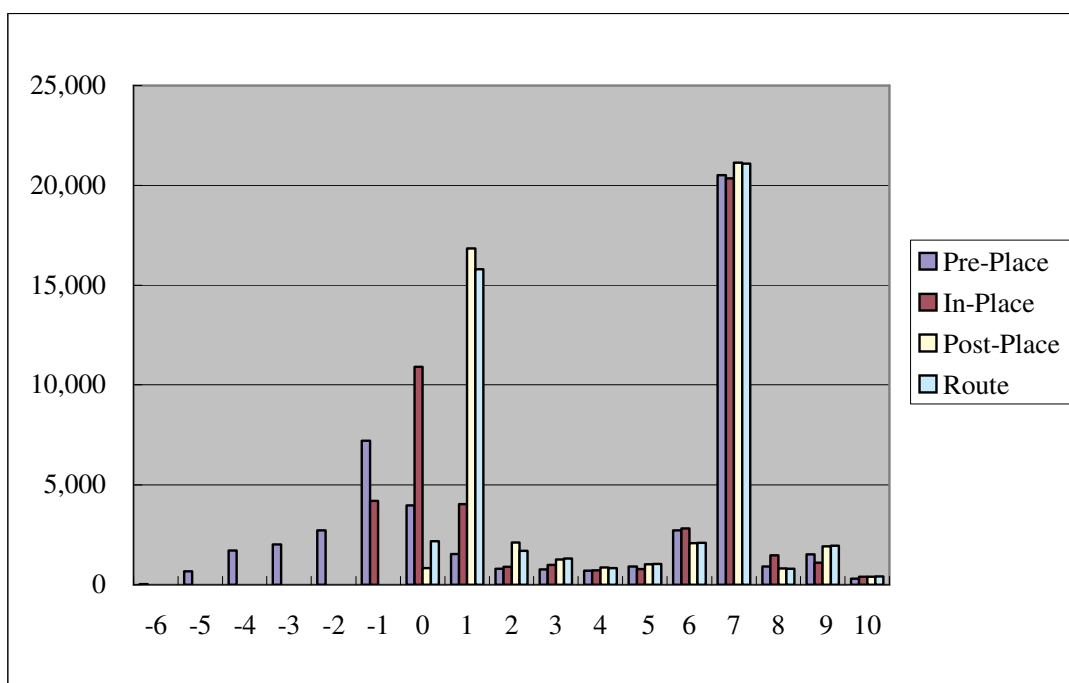
(A) APOLLO

speed	Edge	Average(%)	$\sigma$	Largest Delay Overestimation(ns)	Largest Delay Underestimation(ns)
MIN	LH	98.080	0.057	0.381	0.197
	HL	97.660	0.056	0.356	0.144
TYP	LH	98.500	0.043	1.393	0.481
	HL	98.500	0.043	1.393	0.481
MAX	LH	98.500	0.043	1.393	0.481
	HL	99.360	0.043	1.932	0.357

(B) ASTRO



(A) APOLLO



(B) ASTRO

Figure 5-1 Histogram of Timing Slack



## 6. REMAINING ISSUE USING LOGIC BIST

Due to the increasing complexity of ASIC design, BIST is a popular testing technique for large and complex designs. BIST has several advantages over other test approaches. For example, with BIST, there is no need to do test generation with ATPG. Also BIST enables at-speed testability.

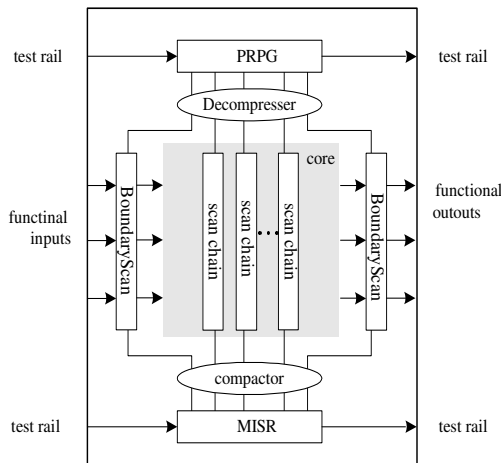
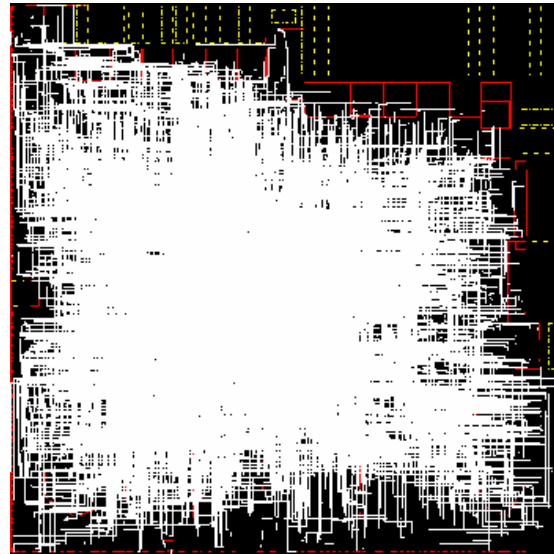


Figure 6-1. STUMPS Architecture

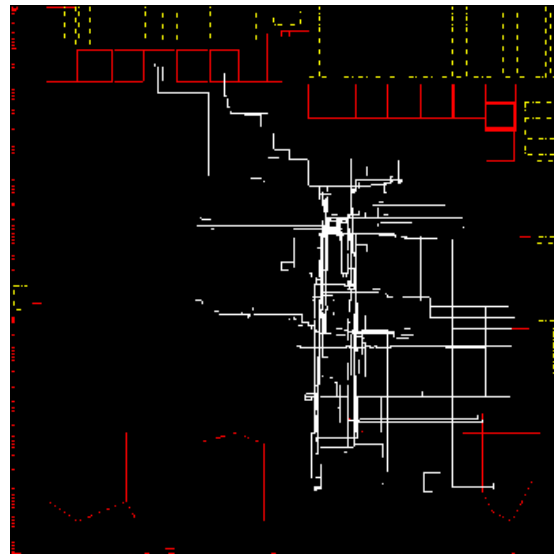
Figure 6-1 shows the well-known STUMPS Architecture [5] scan based BIST approach. Multiple scan paths are driven in parallel by a Pseudo-Random Pattern Generator (PRPG). Test response analysis is done by parallel signature compaction using a Multiple Input Shift Register (MISR). PRPG and MISR are made from a Linear Feedback Shift Register (LFSR) [6]. There is no restriction of number of scan chain from the ATE, and no limit to the number of scan test pins. Thus the scan chain could be divided into several hundreds or more for test cost reduction. On the other hand, the congestion might be worse. It contains a Decompressor, XOR trees as compactor placed between PRPG / MISR, and a great numbers of scan path.

Figure 6-2 shows the routed track's wire for the BIST approach having 2,400 scan paths and 16 scan paths. These figures show the wires of BIST controller including decompressor and compactor, and not including scan path itself. As shown in the figure, the wires are routed across the whole chip in Figure 6-2(A). In fact, congestion is so bad that

we could not finish routing.



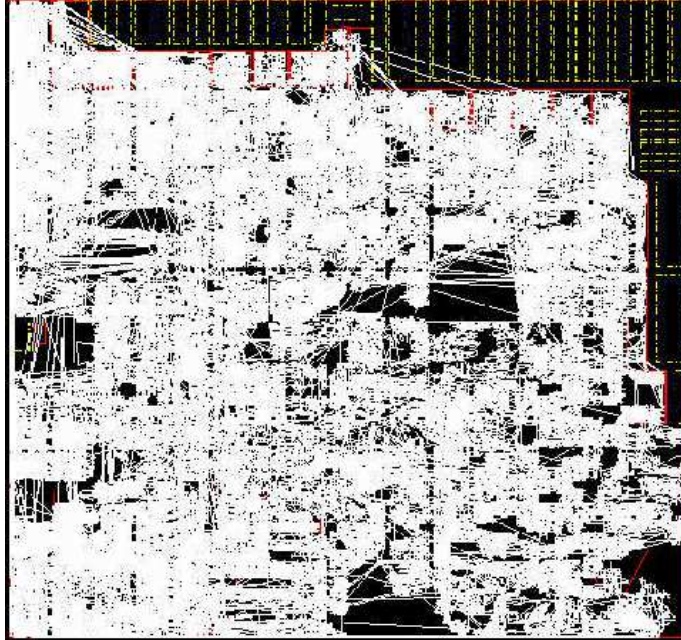
(A) 2,400 scan paths



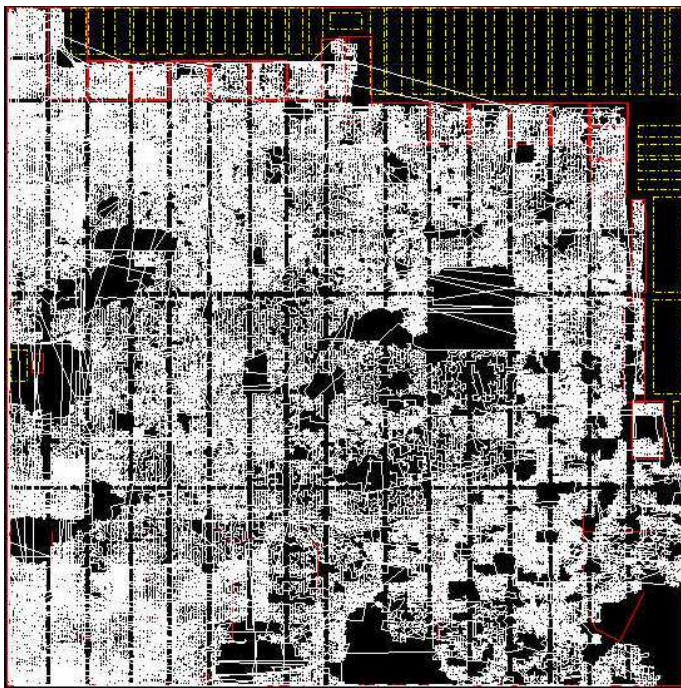
(B) 16 scan paths

Figure 6-2 routed track's wire

Traditionally, the scan path design would be optimized at routing phase by scan reordering process. Figure 6-3 shows the scan paths routing track with and without scan reordering.

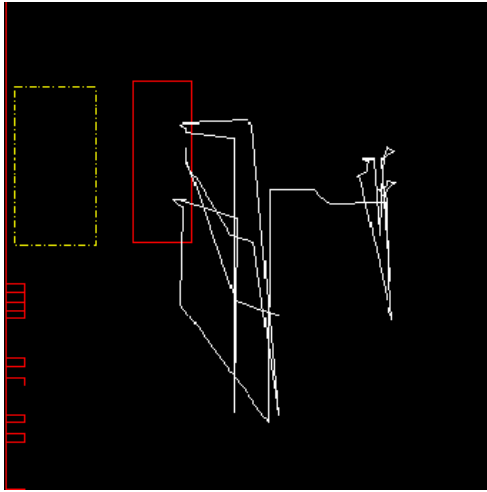


(A) without reordering

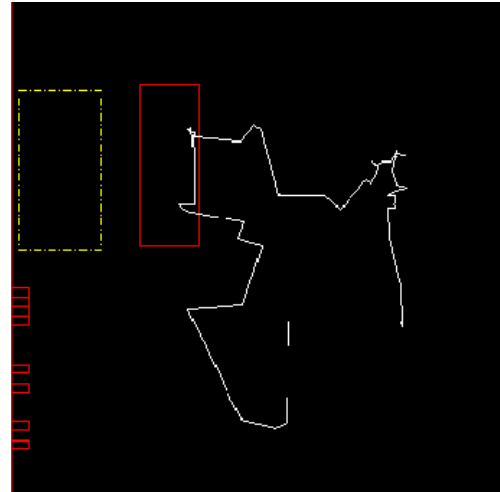


(B) with reordering

Figure 6-3 16 scan paths design

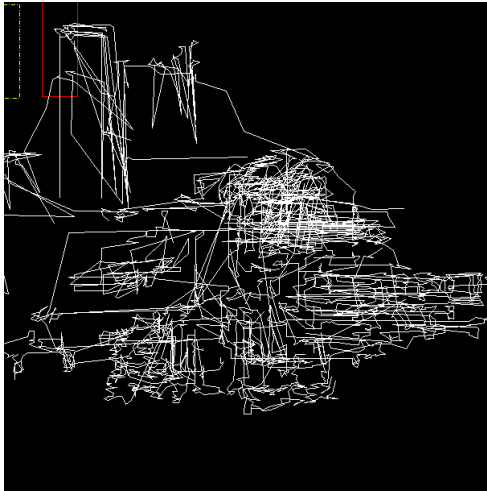


(A) without reordering



(B) single scan path reordered alone

Figure 6-4 one of 2400 scan paths reordered individually



(A) without reordering



(B) 50 scan paths reordered together

Figure 6-5 50 of 2,400 scan paths reordered together

Figure 6-4 shows a single scan path with and without reordering. It seems that scan chain reordering works fine. When we reordered 50 scan paths together, Figure 6-5 shows that congestion is not so improved.

This is because the current scan chain reordering cannot move flip-flops from one scan chain to another. We propose the necessity of the improvement of reordering process across the different scan chains shown in Figure 6-6.

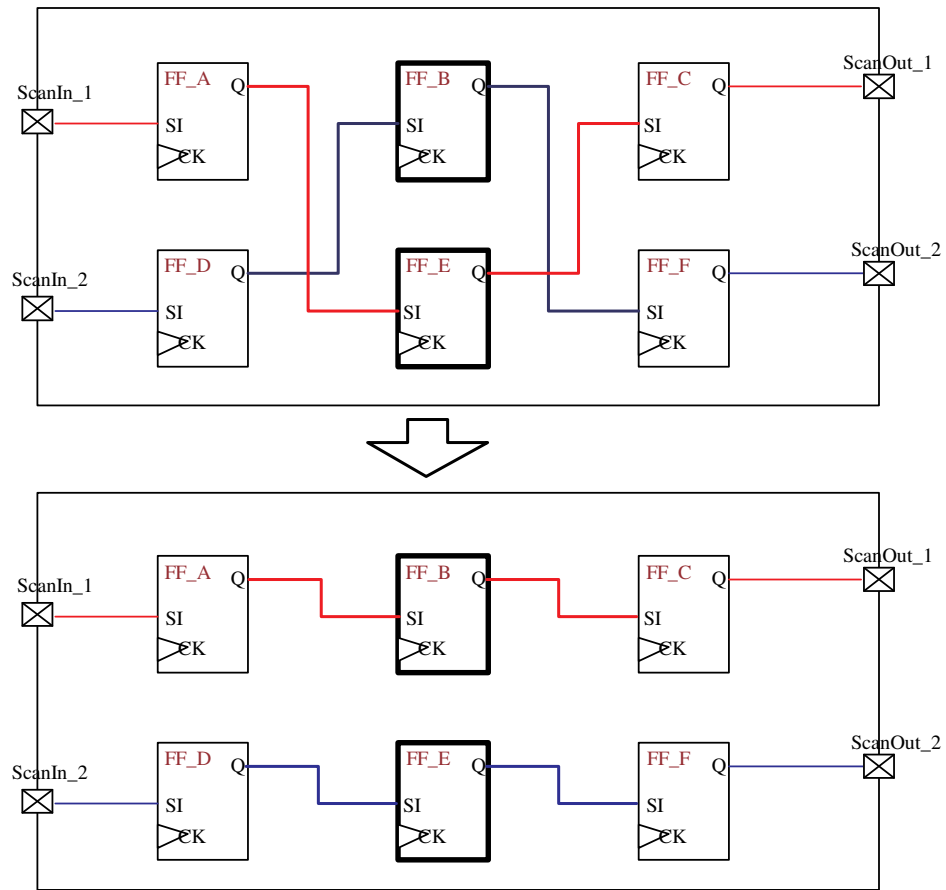


Figure 6-6 Example of reordering across the different scan chains

## 7. CONCLUSIONS

As mentioned above, 5 million gate ASIC can be practically designed by ASTRO using a flat layout style. But APOLLO/SATURN's maximum gate size is about 3 million. We strongly recommend using ASTRO for designing over 3million gate ASIC in flat layout style, or you will not be able to get timing convergence. Our next interest concerns designs over 5 million. 0.13  $\mu$ m processes enable us to integrate 10 million gates. We continue to do the benchmark of 7.5, 10million gate test with ASTRO. Maybe we will be able to present these results of ASTRO at AURORA.

Also we pointed out the remaining issue regarding Logic-BIST. About over 10million gate ASIC' case, testing time will become a big problem. To reduce testing time, we must divide scan chains into many smaller chains. But current scan chain reordering capability is just applicable for each scan chain. In ASICs with many smaller scan chains, such as implemented Logic-BIST, congestion is not improved by the current reordering capability. In the future, the capability of reordering across the different scan chain will be indispensable.

Finally, we would like to mention concurrent place and clock tree synthesis. This function is not implemented yet, but we think this is very important to shorten turn around time. We expect early release of this capability.

## 8. ACKNOWLEDGEMENTS

Would like to acknowledge the Maingate Electronics people, especially Isao Okura, Yoshihiro Ueda.

## 9. REFERENCE

- [1] Jens Vygen - Univ. of Bonn, Bonn, Germany: ALGORITHMS FOR LARGE-SCALE FLAT PLACEMENT,, Proc. of the 34th Design Automation Conference, June 1997, pp. 746.
- [2] Charles J. Alpert, Jen-Hsin Huang, Andrew B. Kahng: Multilevel Circuit Partitioning,, Proc. of the 34th Design Automation Conference, June 1997, pp. 530.
- [3] Hsiao-Pin Su, Allen C.-H. Wu, Youn-Long Lin: A Timing-Driven Soft-Macro Resynthesis

Method in Interaction with Chip Floorplanning,, Proc. of the 36th Design Automation Conference, 1999, pp. 262-267.

- [4] Jacques Benkoski, Andrzej J. Strojwas: The Role of Timing Verification in Layout Synthesis,, Proc. of the 28th Design Automation Conference, 1991, pp. 612-619.
- [5] P. H. Bardell and W. H. McAnney, "Self-Testing of Multichip Modules," Proc. of the IEEE International Test Conference, 1982, pp. 200-204.
- [6] P.H. Bardell, W.H. McAnney, and J. Savir. Built-In Test for VLSI: Pseudorandom Techniques. John Wiley and Sons, New York, 1987.