International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)
Volume 3, Issue 2, February 2014

PERFORMANCE OPTIMIZATION USING DELAY PADDING STRATEGY IN FIELD PROGRAMMABLE GATE ARRAYS

S.Kanaga[1], Mrs.S.Thilakavathi[2]
[1]M.E VLSI Design ,Vivekanandha College of Engineering For Women
[2]AP/ECE, Vivekanandha College of Engineering For Women

Abstract— **Now a days, an Electronic circuits are designed by using EDA tools. This can be implemented in Field Programmable Gate Arrays (FPGA). In FPGA, the pulsed latches are replaced by flip-flops for timing optimization. Pulsed latches are transparent latches driven by a clock with a non standard duty cycle. Latches are used for timing optimization and it can avoid the power and area utilization. This can be achieved by using Common Clock Method to reduce the power and timing constraints in different flip flops. In level triggering, certain flip -flops are replaced with latches for performance gain, but it will increasing the delay in short paths. Comparing with level triggering, the edge triggering perform faster in well defined moment in time. In FPGA performance can be improved by applying Common Clock and Delay Padding can be optimized.**

***Index Terms*-Field Programmable Gate Array, Delay Padding, Timing Analysis.**

I. INTRODUCTION

FPGAs are programmable digital circuits that allow the implementation of a wide array of digital designs. The advancement of process technology, architectural and computer- aided design (CAD) has allowed FPGAs to be a viable platform for an ever increasing number of applications. Unlike Application Specific Integrated Circuits (ASICs), FPGA allows for rapid design prototyping, incremental design debugging, and also avoid high nonrecurring engineering costs. Unfortunately, the advantages of programmability come at a price for area, performance and power consumption. The FPGA designs [1] require 20-40x
more area, 12xmore dynamic power, and 3-4 x slower than their equivalent ASIC implementation.

Our work explores FPGA designs which can be made to run faster by automatically converting a flip-flop based design to use a mix of flip-flops and level-sensitive latches. Level sensitive latches achieve time borrowing by providing a window of time in which signals can freely pass through. The timing analysis using latches is more difficult because transparency allows critical long (Max delay) and short (Min delay) paths to extend across multiple combinational stages. Using pulsed latches driven by a clock with a non standard duty cycle or pulse width is one method of reducing the effects of short paths plaguing conventional latch based circuits, while allowing time borrowing for long paths. This is a best option as commercial FPGAs can generate clocks with different duty, as well as allow the sequential elements to be used as either flip-flops or latches [2],[3]. Converting of flip flop to latch does not involve a hardware change; rather , the prefabricated storage elements on the FPGA can act as either latches or flip-flops, depending on the programming of SRAM configuration cells internal to the FPGA.

To recognize is that commercial FPGAs already contain pulsed the necessary hardware functionality to support pulsed latch-based timing optimization.

II. CONTRIBUTION

The motivation of this paper is to explore using pulsed latches for timing optimization in FPGAs[4]. The selectively insert latches into already routed flip-flop based designs for improve timing performance without extra clocks or logic. The performance improvement succeed by clock skew with common clock. To explore different methods of increasing the delay of short paths for performance improvement. The use of flip-flop to avoid fixing the majority of short path violations caused by the transparency nature of latches. Clocked devices are specially designed for synchronous systems; such devices ignore their inputs except at the transition of a dedicated clock signal (known as clocking, pulsing, or strobing). Clocking causes the flip-flop to either change or retain its output signal based upon the values of the input signals at the transition. Some flip-flops change output on the rising edge of the clock, others on the falling edge.

III.METHODOLOGIES

A.COMMON CLOCK METHOD

We use common clock for all flip-flops to reduce the power ,area and time constraints. The Pulsed latches are replacing by a certain flip flops in field-programmable gate arrays (FPGAs) for reduction of power and time. Pulsed latches are transparent latches driven by a clock with a nonstandard duty cycle. Using edge triggering way to determine pulse width. In edge triggering the circuit becomes active at negative or positive edge of the clock signal. The two ways of edge triggering in this circuit: First is positive edge triggering, Next is negative edge triggering. The circuit is positive edge triggered, it will take input at exactly the time in which the clock signal goes from low to high. Similarly input is taken at exactly the time in which the clock signal goes from high to low in negative edge triggering. In this way we can reduce short and long path delays.

ISSN: 2278 – 909X

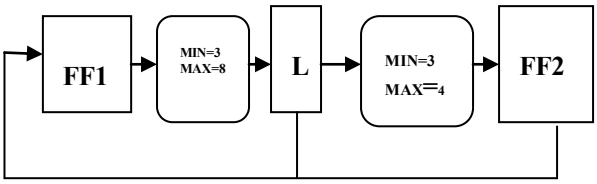
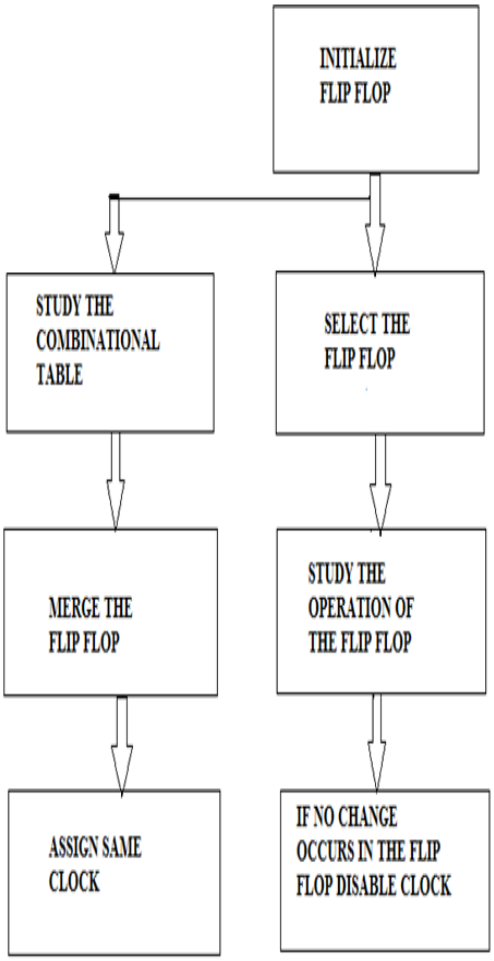


Fig 1.Latch replacement with delay

In this outline the first step for intializing the flip-flop with their different types JK FF,D FF,T FF,SR FF.Then next select the flip-flop for this method and determine which flip-flop for most suitable with their operatins. If no change occurs in the outputs means the clock disabled.In other way of outline using combinational table helps to determine the short path and long path violations. Here, merge the flip-flop into different positions and assign same clock for all flip-flops.every flip flops are connected one by one and depends on the inputs.



B. TIMING ANALYSIS

Flip-flop based circuits has setup-time and hold-time constraints. Setup time is the minimum amount of time the data signal should be held steady before the clock event. So, that the data are reliably sampled by the clock. This applies to synchronous input signals to the flip-flop. Hold time is the minimum amount of time and the data signal should be held steady after the clock event. So, that the data are reliably sampled. This also applies to synchronous input signals to the flip-flop.

$$T_{cq} + CD_{ji} + T_{su} \leq P \text{ , } \forall j \quad i \qquad (1)$$

$T_{cq,}$ or clock-to- Q time .accounts for the lag time between the output of a flip-flop reacting to its input after a clock event. CD_{ji} is the maximum combinational delay of any path starting and ending at flip-flops j and i , respectively. T_{su} represents the flip-flop's setup time. Synchronous sequential circuit.

obey the hold time constraint. Then the any one of the flip-flop output get corrupted. So, the following equation helps to prevent this situation,

$$T_{cq} + cd_{ji} \geq T_{h}, \forall j \quad i \qquad (2)$$

Cd_{ji} represents the delay between j and i. T_h also known as the hold time. Failure to meet hold-time constraints results in a circuit that malfunctions with any value of clock period.

C. CLOCK SKEW

The difference between arrival times of the clock at different devices is called clock skew.The clock skew can be used as a manageable resource and help reduce the clock period in [9].

For,FPGAs, shifted clock lines provide over A 20% improvement in circuit speed[11].as clock comprise ~ 19% of dynamic power consumption in commercial FPGAs[12]. The related work presented in [13] used PDEs on the clock tree,where as the PDEs were inserted into FPGA logic elements in[14].

D. RETIMING

Retiming is the technique of moving the structural location of latches or registers in a digital circuit to improve its performance, area, and/or power characteristics in such a way that preserves its functional behavior at its outputs. Retiming was first described by Charles E. Leiserson and James B. Saxe[10], retiming using level-sensitive latches[17] ,and retiming used for low power[12],[13].

Retiming changes the position and number of flip-flops,making the design debugging process more difficult as a designer.time-borrowing via retiming is inherently quantized ,because it is impossible to relocate a flip-flop.retiming has been applied to FPGAs [18], linear time algorithm that provides a 7% improvement in circuit speed.

III.EDGE TRIGGER LATCHES

The input signal is sampled at the rising edge or falling edge of the clock signal. It is not sensitive to Glitches (example: Flip flop). Edge-triggering is good for clocks, because it allows the value output by a latch in response to one (e.g. rising) clock edge to be used in the computation of what it should do on the next rising clock edge. Less complex clocking with edge triggered devices

The transparent nature of level-sensitive latches allow signals of one combinational stage to arrive transparent phase of the next phase cycle.the advantage of using level sensitive latches is avoid the dynamic power consumption overhead of using multiple clocks.It is implemented in clock

ISSN: 2278 – 909X

skew and possible for increasing in the number of flip-flops if retiming is used.Because,the transparent latches use less power than master –slave flip-flops,however unlike flip-flops,they do not filter glitches.latch-based optimization does not change the locations of storage elements in the netlist.

TABLE I
COMBINATIONAL TRUTH TABLE

D3	D2	D1	D0	S1	S2	S3	S4
0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0
0	0	1	0	1	0	0	0
0	0	1	1	0	1	0	0
0	1	0	0	1	0	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	0	0	1
1	0	0	0	1	0	0	0
1	0	0	1	0	1	0	0
1	0	1	0	0	1	0	0
1	1	0	0	0	1	0	0
1	1	0	1	0	0	1	0
1	1	0	1	0	0	1	0
1	1	1	0	0	0	1	0
1	1	1	1	0	0	0	1

There are two notable differences between transparent latch and positive edge triggered flip-flop timing parameters.

1) T_{su} and T_{hold} are bound to the falling edge of the clock rather than the rising edge.
2) T_{dq} represents the data-to-Q time ¬the time lag between the output of a flip-flop to a changeable input.

Level-sensitive latches allow signals to arrive at any time before the T_{su} timing window.this latches driven by a single clock can mimic multiple skews.The time borrowing properties of latches have definite advantages.Minmum delay sequential circuit having hold-time violations.So, we can reduce the size of the transparent window , possible to avoid such hold-time violations.

Pulsed latches are widely used in microprocessors for better performance[14].Their use for improving the performance of ASICs recently explored [15],[16].

IV.DELAY PADDING

Minimum delay constraints can always be satisfied by delay padding, maximum delay constraints may not. For using the delay padding the possiability of increasing the pulse width for more time borrowing opportunities. we are fixing the short path violations by increasing their path delays by taking a more circuitous route in the router- delay padding. The process of determining which combinational paths to pad for a wider pulse width. It is a side effect of our latch-based optimizations.the delay padding in FPGA routing to correct hold time violations,in this paper applies specifically in the latch-based optimization.

The delay padding it could be made more effective in this ways: A pulse width selection scheme that adapts to a benchmark's short path delay distribution, The delay padding strategy should be a hybrid of the two separate strategies we attempted. That is, use free routing resources whenever possible and only rip-up and re-route what is necessary if routing blockages are present.

V.RESULTS

This combinational table shows that the operation behind clock connected to every flip-flops one by one respectively.

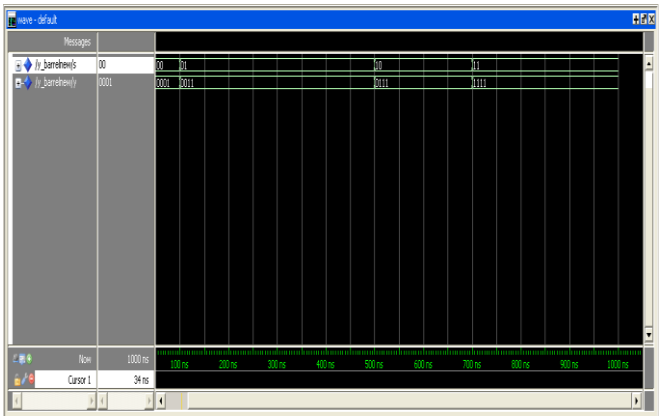


Fig 3. Output for Disable Clock

In this result shows,without using any clocks the input will be triggered for negative and positive edges.

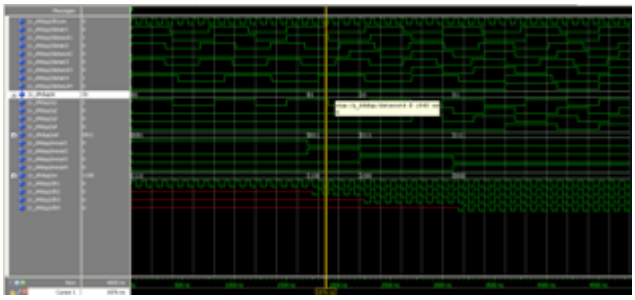


Fig 4. Output for Common Clock for flip flops

In the above result shows. the clock connected to every flip-flop one by one respectively. In this way, the power, area and time reduced.

V. CONCLUSION

This paper, the method used is applied to the FPGA architecture for using different flip flop for reducing power, delay using common clock method. In future work Clock Gating Method and Clock Tree Method will be used and compared with this method. It helps to increase the performance in the FPGA architecture.

VI. ACKNOWLEDGMENTS

Our sincere thanks to the friends for the helpful discussions and valuable suggestions about the presentation of this work.

VII.REFERENCES

[1] I. Kuon and J. Rose, "Measuring the gap between FPGAs and ASICs," in *Proc.ACM FPGA*, 2006, pp. 21–30.
[2] *Virtex-6 FPGA Configurable Logic Block*, Xilinx, Inc., San Jose, CA, 2009.

ISSN: 2278 – 909X

[3] *Virtex-6 FPGA Clocking Resources*, Xilinx, Inc., San Jose, CA, 2011.
[4] B. Teng and J. H. Anderson, "Latch-based performance optimization for FPGAs,in *Proc. IEEE FPL*, Sep. 2011, pp. 58–63.
[5] J. P. Fishburn, "Clock skew optimization," *IEEE Trans. Comput.*, vol. 39, no. 7,pp. 945–951, Jul. 1990.
[6] D. P. Singh and S. D. Brown, "Constrained clock shifting for field programmable gate arrays," in *Proc. ACM FPGA*, 2002, pp.121–126.
[7] T. Tuan, A. Rahman, S. Das, S. Trimberger, and S. Kao, "A 90-nm low-powerFPGA for battery-powered applications," *IEEE Trans. CAD*, vol. 26, no. 2, pp. 296–300, Feb. 2007.
[8] C.-Y. Yeh and M. Marek-Sadowska, "Skew-programmable clock design for FPGA and skew-aware placement," in *Proc. ACM FPGA*, 2005, pp. 33–40.
[9] X. Dong and G. Lemieux, "PGR: Period and glitch reduction via clock skew scheduling, delay padding and GlitchLess," in *Proc. IEEE FPT*, Dec. 2009, pp. 88–95.
[10] C. Leiserson and J. Saxe, "Retiming synchronous circuitry," *Algorithmica*, vol. 6, no. 1, pp. 5–35, 1991.
[11] B. Lockyear and C. Ebeling, "Optimal retiming of level-clocked circuits using symmetric clock schedules," *IEEE Trans. CAD*, vol. 13, no. 9, pp. 1097–1109, Sep. 1994.
[12] J. Monteiro, S. Devadas, and A. Ghosh, "Retiming sequential circuits for low power," in *Proc. IEEE/ACM ICCAD*, Nov. 1993, pp. 398–402.
[13] K. Lalgudi and M. Papaefthymiou, "Fixed-phase retiming for low power design," in *Proc. IEEE ISLPED*, Aug. 1996, pp. 259–264.
[14] H. Ando, Y. Yoshida, A. Inoue, I. Sugiyama, T. Asakawa, K. Morita, T. Muta, T. Motokurumada, S. Okada, H. Yamashita, Y. Satsukawa, A. Konmoto, R. Yamashita, and H. Sugiyama, "A 1.3 GHz fifth generation SPARC64 microprocessor," in *Proc. IEEE ISSCC*, vol. 1. 2003, pp. 246–491.
[15] H. Lee, S. Paik, and Y. Shin, "Pulse width allocation and clock skew scheduling: Optimizing sequential circuits based on pulsed latches," *IEEE Trans. CAD*, vol. 29, no. 3, pp. 355–366, Mar. 2010.
[16] S. Paik, S. Lee, and Y. Shin, "Retiming pulsed-latch circuits with regulating pulse width," *IEEE Trans. CAD*, vol. 30, no. 8, pp. 1114–1127, Aug. 2011.
[17] R. Fung, V. Betz, and W. Chow, "Slack allocation and routing to improve FPGAtiming while repairing short-path violations," *IEEE Trans. CAD*, vol. 27, no. 4,pp. 686–697, Apr. 2008.
[18] .P.Singh,V.Manoharajah, and S.d. Brown,"Incremental retiming for FPGA physical synthesis," in *Proc. IEEE/ACM DAC*, Jun. 2005, pp. 433–438.

ISSN: 2278 – 909X