RESEARCH ARTICLE OPEN ACCESS

Power Optimization Technique Based On Multi-Bit Flip-Flop Design

P. Sathyaa, K. Sinduja
M.E. VLSI DESIGN Coimbatore Institute of Engineering and Technology, Narasipuram, Coimbatore.
Asst. Prof/Dept of ECE Coimbatore Institute of Engineering and Technology, Narasipuram, Coimbatore.

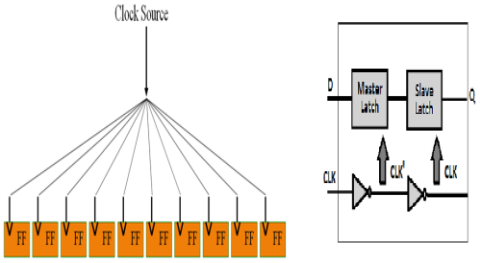
Abstract
Reducing power plays a vital role in low power VLSI design. In the VLSI design circuits, clocking is the most dominating power consuming element. To acquire the optimum power, it is essential to reduce the clocking power. Hence this paper describes a method for reducing the power consumption by replacing some flip flops with fewer multi-bit flip-flops. A combination table is used here to store the possible combination of flip-flops, for reducing the search space and can be merged. This concept is implemented in a simple memory circuit. Our proposed work will reduce the power by triple the time than that of the conventional system.

I. Introduction

Nowadays, achieving low power consumption is a tedious one in IC fabrication industries. As well as it needs a faster clock which consumes more power. Clock tree synthesis (CTS) is the process of insertion of buffers or inverters along the clock path of ASIC design, in order to achieve zero/minimum skew or balanced skew Large clock skew can force the designer to use large time period between clock pulses. This makes the system slower. Usually, clock trees are created by using buffers or inverters. Building a inverter clock tree would use less FFT's than building buffer clock tree to urge a similar target skew. Thus, it might get less delay and power. To optimize the power consumption, several low-power design techniques are introduced, such as clock gating, substituting non-timing-critical cells with their high-Vt counter components, power gating, making multi-supply-voltage styles, dynamic voltage/ frequency scaling and minimizing clock network.. Among these techniques, minimizing clock network is extremely vital in reducing power consumption as a result of it accounts for up to 45% of dynamic power of the chip. In this clock system power, 90% is consumed by the flip-flops themselves and also the last branches of the clock distribution network that directly drives the flip-flops. This is often due to the high switching activity. Unremarkably, single bit flip-flop are used and every flip-flop needs a separate clock buffers, which consumes large power. The rest of this paper is structured as follows. Section 2 describes the conventional flip-flop and their problems. In Sections 3 will explains the proposed method and section 4 describes the implementation details. Section 5 and Section 6 describes the simulation analysis and conclusion respectively.

II. Conventional System

A flip-flop is a single bit memory storage element, which is very similar to a latch in that it is a bistable multivibrator, having two states and a feedback path that allows it to store a single bit of information. The difference between a latch and a flip-flop is that a latch is level triggered and the flip-flop is edge triggered. There are several different types of flip-flop each with its own uses and peculiarities. The four main types of flip-flop are: SR, JK, D, and T.
In conventional type, In Fig: 1(a) each flip-flop requires a separate clock source which increase the clocking power considerably also the chip area, voltage swing and clock skew will get increased.



(a) (b)

Fig:1(a) A path from clock source to clock sinks(flip-flops),(b) single bit flip-flop

Fig: 1(b) shows a single-bit flip-flop, that has two latches (Master latch and slave latch). The latches need "Clk" and "Clk' " signal to perform the operations. In order to have better delay from Clk-> Q, we will regenerate "Clk" from "Clk' ". Hence we will have two inverters in the clock path. The simulation results of power consumption of D, JK,

RS, and T flip-flops are shown in Fig:2 the power is analysed by Xilinx ISE Design suite 12.3.

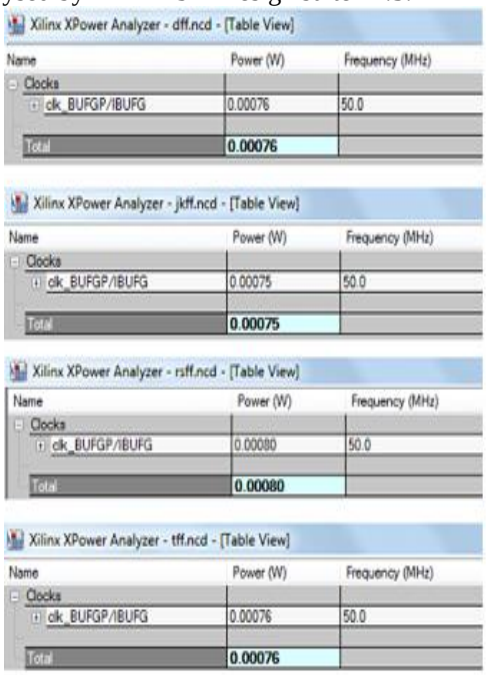


Fig: 2 Simulation results of Power consumption of D, JK, RS, & T flip-flop

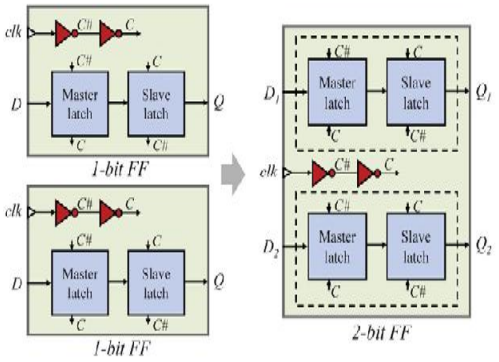
III. Proposed System

The aim of the present work is therefore to propose the Multi-bit flip-flops which merge the single bit flip-flop that share the clock buffer. As CMOS technology progresses, the driving capability of an inverter-based clock buffer increases significantly. The driving capability of a clock buffer can be evaluated by the number of minimum-sized inverters that it can drive on a given rising or falling time. Due to manufacturing ground rules inverters in flip-flops tend to be oversized. As we get into smaller geometries like 65nm and beyond, the minimum size of clock driver can drive more than single flip-flop.

MBFFs may offers some of the advantages is as follows:

1) design area is being small;
2) needs less power and delay;
3) controllable clock skew;
4) routing resource utilization is improved;

According to section 2, among those flip-flops, D flip-flop is chosen for this project because of its simplicity and low power consumption.Fig:3 shows the merging of 2 single bit flip-flops.



(a) (b)

Fig: 3 Merging of flip-flops (a) 2 single bit flip-flop(before merging) (b)2 bit flip-flop(after merging)

In this section, the multi-bit flip-flop concept is introduced. Before that, we will review single-bit flip-flop Fig: 3 shows an example of merging two 1-bit flip-flops into one 2-bit flip-flop. If we replace the two 1-bit flip-flops as shown in Fig. 3(a) by the 2-bit flip-flop as shown in Fig. 3(b), the total power consumption can be reduced because the two 1-bit flip-flops can share the same clock buffer. Merging single-bit flip-flops into one multi-bit flip-flop can avoid duplicate inverters, and lower the total clock dynamic power consumption. The total area contributing to flip-flops can be reduced as well.

Fig: 4 shows the flow graph of our proposed work. Our proposed work is roughly divided into three stages. Identifying merge able flip-flops; Build a combination table; and Merge flip-flops. In first stage, we have to identify the flip-flops which are supposed to be combined. In second stage, a combination table can be formed, which defines all possible combinations of flip-flops in order to get a new multi-bit provided by library. In third stage, the flip-flops can be merged with the help of the table.

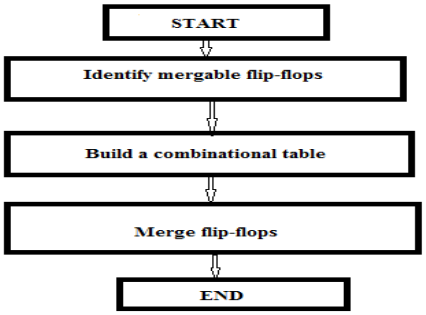


Fig: 4 Flow graph of the proposed system.

3.1 Identifying Merge able Flip-Flops

Multi-bit flip-flop cells are capable of decreasing the power consumption because they have shared inverter inside the flip-flop. Meanwhile, they can minimize clock skew at the same time.

To obtain these benefits, the ASIC design must meet the following requirements. The single-bit flip-flops, thus we want to replace with multi-bit flip-flops must have same clock condition and same set/reset condition.

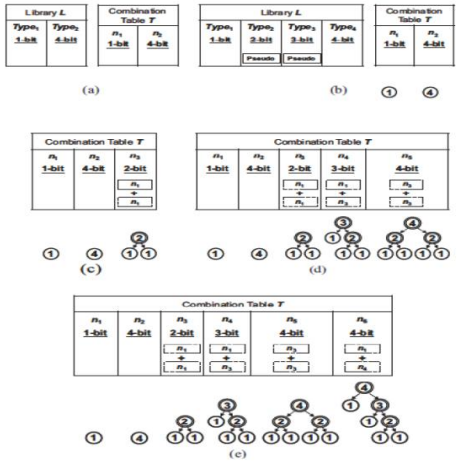
3.2 Build a combination table

If we wish to interchange many flip-flops by a new flip-flop, we have to make sure that the new flip-flop is provided by the library L, when the possible regions of those flip-flops overlap. Fig: 5 shows the instance of combination table.

In this paper, we'll build a combination table, that records all attainable combinations of flip-flops to get feasible flip-flops before replacements. Thus, we are able to do bit by bit replacement of flip-flops consistent with the order of the combinations of flip-flops in this table. Since just one combination of flip-flops has to be thought of in every time, the search time are often reduced greatly. In this segment, building a combination table is illustrated.

A binary tree is employed to represent one combination for simplicity. Every node within the tree denotes one sort of a flip-flop in L. The types of flip-flops denoted by leaves can represent the sort of the flip-flop within the root. For every node, the bit width of the corresponding flip-flop equals to the bit width summation of flip-flops denoted by its left and right child.

Let ni denote one combination in T. In the beginning, we tend to initialize a combination ni for every kind of flip-flops in L. Then, so as to represent all combinations by employing a binary tree, pseudo types could value-added, which denotes the flip-flops that are not provided by the library. As an example, assume that a library solely supports 2 varieties of flip-flops whose bit widths are one and four, respectively. So as to use a binary tree to denote a combination whose bit width is four, there should exist flip-flops whose bit widths are 2 and 3. Thus, we've to form 2 pseudo styles of flip-flops with 2 and 3-bit, if L doesn't offer these flip-flops.



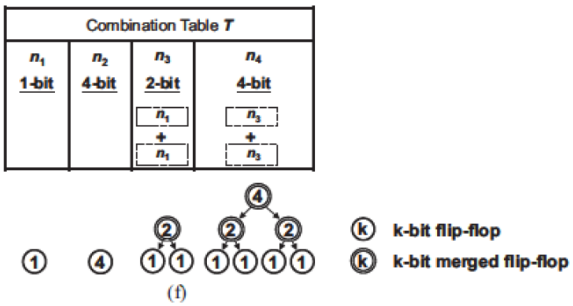


Fig: 5 Example of building the combination table. (a)The library and combination table (b) Insertion of pseudo type (c) combine two n1s and form n2. (d) Combining n1 and n3, n5 is formed . (e)Combining n1 and n4, n6 is obtained (f) Final combination table.

Let bmax and bmin denote the maximum and minimum bit width of flip-flops and inserts all flip-flops whose bit widths are larger than bmin and smaller than bmax. For each types of combinations in T, attempt to combine them for making a new combination. If the new combination of the flip-flop is possible, then we might add it to the table T.

Now, for every combination in T, a binary tree would designed with 0-level, and therefore the root of the binary tree denotes the combination. Next, attempt to build new legal combinations consistent with the current combinations. By combining 2 single-bit flip-flops within the initial combination, a new combination n3 with 2-bit flip-flop will be obtained. Similarly, we are able to get a new combination n4 (n5) by combining n1 and n3 (two n3's). Finally, n6 is obtained by combining n1 and n4.

All attainable combinations of flip-flops are shown in Fig. 9(e). Among these combinations, n5 and n6 are duplicated since they each represent an equivalent condition, that replaces four 1-bit flip-flops by a 4-bit flip-flop. To speed up the program, n6 is deleted from T instead of n5 as a result of its height is larger. After this procedure, n4 becomes an unused combination since the foundation of binary tree of n4 corresponds to the pseudo type, type3, in L and its solely enclosed in n6. When deleting n6, n4 is additionally got to be deleted.

In order to specify all attainable combinations within the combination table, all the flip-flops whose bit widths vary between bmax and bmin and don't exist in L ought to be inserted into L within the above procedure. However, this is often time overwhelming. To enhance the period, just some sorts of flip-flops ought to be inserted.

3.3 Merging of Flip-Flops

A combination table is built in section 3. Now, we use the combination table to combine flip-flops will be explained in this subsection.

We may have so many combinations to form a single multi-bit flip-flop. For example, consider that we have to build a 4-bit flip-flop, we may have 3 combinations are there to setup that, i.e. by merging four 1-bit flip-flops else by merging two 2-bit flip-flops or else it can be formed by merging one 3-bit and one 1-bit flip-flop.

Among those combinations, we've got to choose the one, that offers low cost. In Fig. 7(a), f1 and f2 are chosen owing to its low cost. Hence, add a replacement node f3 within the list below n4, so delete f1 and f2 from their original list. Similarly, f4 and f5 are combined to get a new flip-flop f6, and therefore the result's shown in Fig.7(c).Finally flip-flops within the combinations of 1-level trees (n4 and n5) are obtained as shown in Fig.7(d), begin to create the flip-flops within the combinations of 2-level trees (n6, and n7). In Fig. 7(e), there exist some flip-flops within the lists below n2 and n4, and that we can merge them to get flip-flops in n6 and n7, respectively. Suppose there's no overlap region between the few flip flops in n2 and n4. It fails to make a 4-bit flip-flop in n6. Since the 2-bit flip-flops f3 and f6 are mergeable, combine them to get a 4-bit flip-flop f10 in n7.

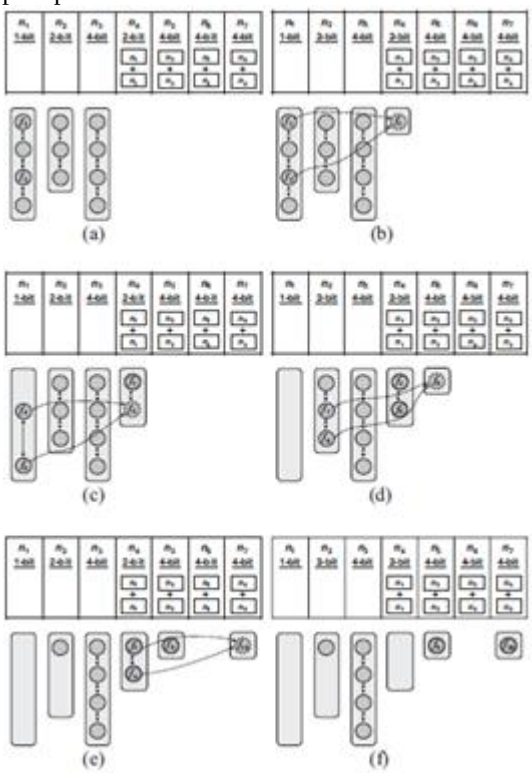


Fig: 7 Example for merging of flip-flops. (a) Flip-flops before merging (b) f1 and f2 are replaced by f3 (c) f4 and f5 are replaced by f6 (d) f7 and f8 are replaced by f9 (e) f3 and f6 are replaced f10 (f) flip-flops after merging.

IV. Implementation Details

We implemented this concept in a simple memory circuit and the power consumption is analysed.

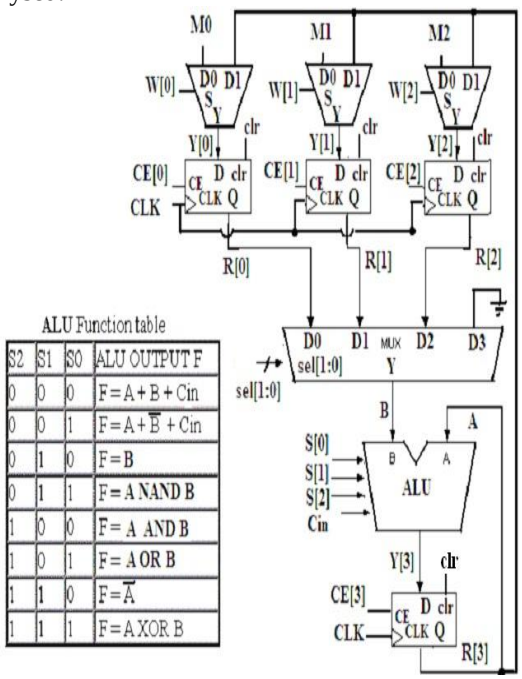


Fig: 8 A simple memory circuit

The above diagram is an example of simple memory circuit with ALU and MUX. Here three single bit flip-flops are used, which requires three separate clock drivers. In our proposed work the multi-bit flip-flop concept is implemented in these flip-flops, which requires single clock driver to drive them and the corresponding power consumption is measured.

V. Result Analysis

We implemented the concept in Verilog language and are executed by Xilinx ISE Design suit 12.3 with Modelsim simulator. The below results shows that the power difference two single bit flip-flops which uses separate clock buffer with the frequency of 50MHZ, consumes 0.00152W power and in proposed system double bit flip-flop,

Fig: 9 Comparison of power consumption between conventional and proposed system using single clock buffer for both the flip-flops with the frequency of 50MHZ, consumes 0.00082W power which is more efficient than the conventional system.

Fig: 10 shows the simulated output for merging of flip-flops i.e. for Fig: 7. Here single clock is used for 7 combinations of flip-flops i.e. from n1 to n7.

Fig.11 and Fig.12 shows the simulated output and power consumption of the simple memory circuit with separate clock.

Fig.13 shows the reduced power consumption of the same circuit but driven by single clock driver.

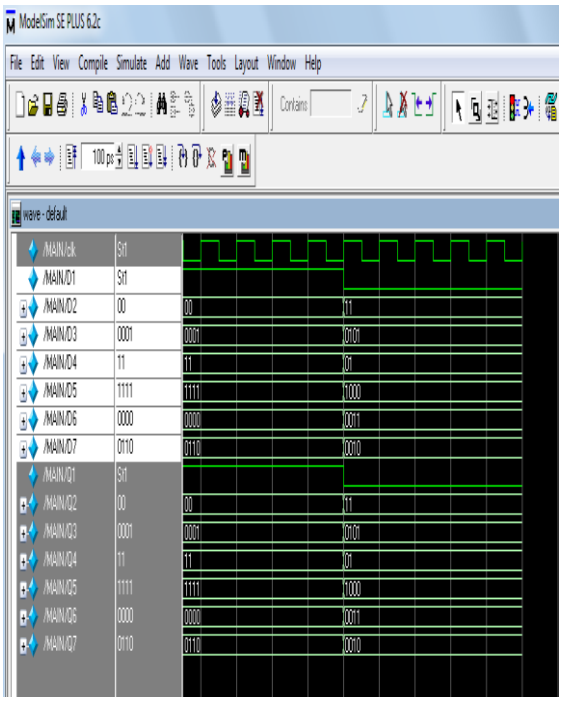


Fig.10 Simulated output for merging of flip-flops

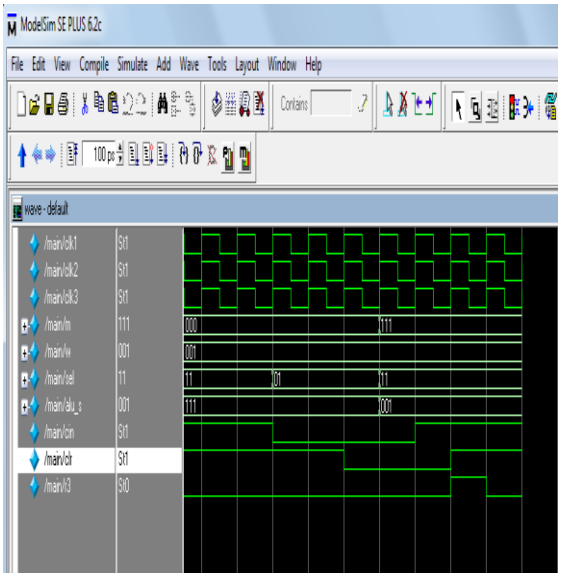


Fig.11 Simulated output for simple memory circuit

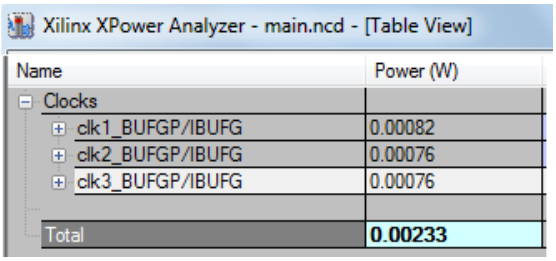


Fig.12 Power consumption of the memory circuit before applying the multi-bit flip-flop concept

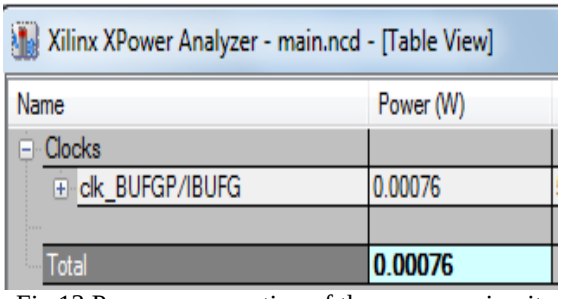


Fig.13 Power consumption of the memory circuit after applying the multi-bit flip-flop concept.

The comparison is made between the power consumption of the memory before merging and after merging. After applying our algorithm, the power is reduced by 67.38% than that of the conventional system.

VI. Conclusion

In this project, the concept of Multi-bit flip-flops is introduced for the replacement of single-bit flip-flop for power reduction in a simple memory circuit design. The procedure of flip-flop replacements is depending on the combination table. Process of identifying the merge able flip-flops and

building a combination table, in which the possible combination of flip-flops are combined together, are executed. Our proposed work will achieve the power reduction by triple the times than that of the conventional system.

References

[1] **"Effective and Efficient Approach for Power Reduction by Using Multi-Bit Flip-Flops,"** Ya-Ting Shyu, Jai-Ming Lin, Chun-Po Huang, Cheng-Wu Lin, Ying-Zu Lin, and Soon-Jyh Chang, *Member IEEE*. IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, VOL. 21, NO. 4, APRIL 2013.

[2] D. Duarte, V. Narayanan, and M. J. Irwin, **"Impact of technology scaling in the clock power,"** in *Proc. IEEE VLSI Comput. Soc. Annu. Symp.*, Pittsburgh, PA, Apr. 2002, pp.s 52–57.

[3] H. Kawagachi and T. Sakurai, **"A reduced clock-swing flip-flop (RCSFF) for 63% clock power reduction,"** in *VLSI Circuits Dig. Tech. Papers Symp.*, Jun. 1997, pp. 97–98. Y. Cheon, P.-H. Ho, A. B. Kahng, S. Reda, and Q. Wang, **"Power-aware placement,"** in *Proc. Design Autom. Conf.*, Jun. 2005, pp. 795–800.

[4] Y. Cheon, P.-H. Ho, A. B. Kahng, S. Reda, and Q. Wang, **"Power-aware placement,"** in *Proc. Design Autom. Conf.*, Jun. 2005, pp. 795–800.

[5] P. Gronowski, W. J. Bowhill, R. P. Preston, M. K. Gowan, and R. L. Allmon, **"High-performance microprocessor design,"** *IEEE J. Solid-State Circuits*, vol. 33, no. 5, pp. 676–686, May 1998.

[6] W. Hou, D. Liu, and P.-H. Ho, **"Automatic register banking for lowpower clock trees,"** in *Proc. Quality Electron. Design*, San Jose, CA, Mar. 2009, pp. 647–652.

[7] Y.-T. Chang, C.-C. Hsu, P.-H. Lin, Y.-W. Tsai, and S.-F. Chen, **"Post-placement power optimization with multi-bit flip-flops,"** in *Proc. IEEE/ACM Comput.-Aided Design Int. Conf.*, San Jose, CA, Nov. 2010, pp. 218–223.