Barre Gnani Ashok* et al.
(IJITR) INTERNATIONAL JOURNAL OF INNOVATIVE TECHNOLOGY AND RESEARCH
Volume No.5, Issue No.2, February – March 2017, 5902-5906.

Implementation of Low Power Delay Pulsed Latches for Shift Register Using KOGGE Stone Adder

BARRE GNANI ASHOK
Kakinada Institute of Engineering and Technology, Yanam Road, Matlapalem, Talarevu Mandal, Corangi, Andhra Pradesh, Pin Code - 533461

THULIMILLI PREM BOSCO
Assistant Professor, Kakinada Institute of Engineering and Technology, Yanam Road, Matlapalem, Talarevu Mandal, Corangi, Andhra Pradesh, Pin Code - 533461

Abstract : **This project proposes delay efficient architecture for shift registers using pulsed latches instead of filp flops. Area and power can be reduced greatly by using latches then flip flops. The timing problem exhibited by the latches is reduced by taking necessary delays in pulses for latches. This includes a counter for generating pulses with delays. For obtaining these delays counter has to incremented by 1. The proposed kogge stone architecture reduces the delay to maximum extent, and produces numerious variation between conventional adder architecture.**

Keywords: **Flip Flops; Latches; Kogge Stone Adder;**

I. INTRODUCTION

Dynamic power is consumed across all elements of a chip. The clock network is one of the large consumers of dynamic power. Therefore, reducing power in the clock network can impact the overall dynamic power significantly. Designers already use a variety of techniques to reduce the clock power using smaller clock buffers. Even with these techniques, the dynamic power of clock network can be large since registers are used as state elements in the design. In general, a flip-flop is used as the register.

A conventional flip-flop is composed of two latches (master and slave) triggered by a clock signal. Flip-flop synchronization with the clock edge is widely used because it is matched with static timing analysis (STA). Timing optimization based on STA is must for SoCs. A methodology has been developed which uses latches triggered with pulse clock waveforms. With this methodology, designers can apply static timing analysis and timing optimization to a latch design while reducing the dynamic power of the clock networks.

II. PULSE LATCH

The pulsed latch requires pulse generators that generate pulse clock waveforms with a source clock. The pulse width is chosen such that it facilitates the transition. The following diagram represents a simple pulse generator and the associated pulse waveform.

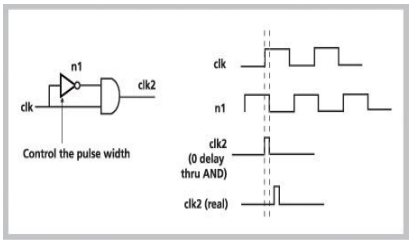


Fig2.1: Pulse Latch

In this methodology, the pulse generators are automatically inserted to satisfy several rules during clock-tree synthesis. Along with pulse generators, this approach also uses a number of matching delay cells to allow for match clock insertion delays with or without pulse generators.

2.1 Shift registers:

Flip flops are use in constructing registers. Register is a group of flip flops used to store multiple bits of data. For example, if a computer is to store 16 bit data, then it needs a set of 16 flip flops. The input and outputs of a register are may be serial or parallel based on the requirement. A shift register is a sequential circuit which stores the data and shifts it towards the output on every clock cycle.Basically shift registers are of 4 types. They are

- Serial In Serial Out shift register
- Serial In parallel Out shift register
- Parallel In Serial Out shift register
- Parallel In parallel Out shift register

2.1.1 Parallel in Serial out shift register:

The input to this register is given in parallel i.e. data is given separately to each flip flop and the output is collected in serial at the output of the end flip flop.

The clock input is directly connected to all the flip flops but the input data is connected individually to each flip flop through a mux (multiplexer) at input of every flip flop. Here D1, D2, D3 and D4 are the individual parallel inputs to the shift register. In this register the output is collected in serial.

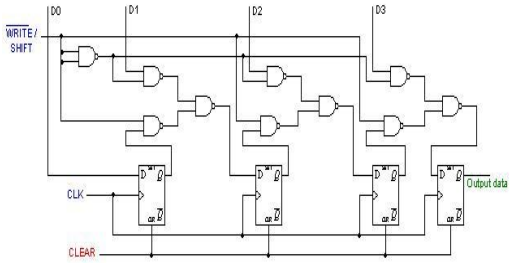


Fig2.2 : Parallel In serial out Shift Register

The output of the previous flip flop and parallel data input are connected to the input of the MUX and the output of MUX is connected to the next flip flop. A Parallel in Serial out (PISO) shift register converts parallel data to serial data. Hence they are used in communication lines where a number of data lines are multiplexed into single serial data line.

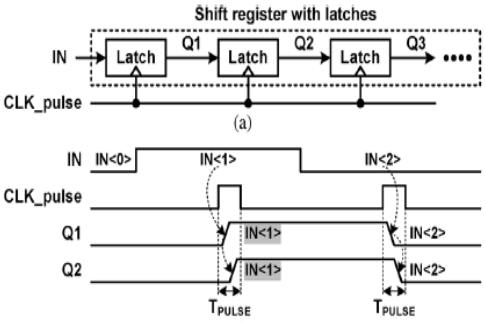


Fig2.3: shift registers with latches and pulsed clock signal

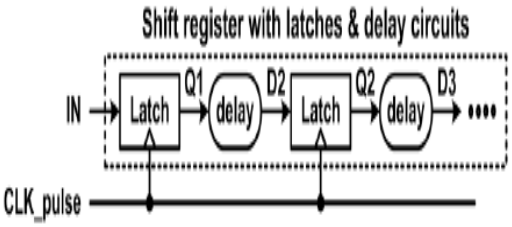


Fig2.4 : Shift Registers With Latches and Delay Circuits

A master-slave flip-flop using two latches can be replaced by a pulsed latch consisting of a latch and a pulsed clock signal. All pulsed latches share the pulse generation circuit for the pulsed clock signal. As a result, the area and power consumption of the pulsed latch become almost half of those of the master-slave flip-flop. The pulsed latch is an attractive solution for small area and low power consumption. The pulsed latch cannot be used in shift registers due to the timing problem. The shift register consists of several latches and a pulsed clock signal (CLK_pulse). The operation waveforms inshow the timing problem in the shifter register. The output signal of the first latch (Q1) changes correctly because the input signal of the first latch (IN) is constant during the clock pulse width . But the second latch has an uncertain output signal (Q2) because its input signal (Q1) changes during the clock pulse width. One solution for the timing problem is to add delay circuits between latches, as shown in Fig. 3(a). The output signal of the latch is delayed and reaches the next latch after the clock pulse. As shown in Fig. 3(b) the output signals of the first and second latches (Q1 and Q2) change during the clock pulse width , but the input signals of the second and third latches (D2 and D3) become the same as the output signals of the first and second latches (Q1 and Q2) after the clock pulse. As a result, all latches have constant input signals during the clock.

III. PULSED LATCH ARCHITECTURE

The power optimization is similar to the area optimization.The power is consumed mainly in latches and clock-pulse circuits.Each latch consumes power for data transition and clockloading. When the circuit powers are normalized with a latch,the power consumption of a latch and a clock-pulse circuit are1 and , respectively. The total power consumption is also. An integer for the minimumpower is selected as a divisor of , which is nearest to .In selection, the clock buffers are not considered.The total size of the clock buffers is determined by thetotal clock loading of latches. Although the number of latchesincreases from to , the increment ratio of theclock buffers is small. The number of clock buffers is , Asincreases, the size of a clock buffer decreases in proportion tobecause the number of latches connected to a clock bufferis proportional to . Therefore, the total sizeof the clock buffers increases slightly with increasing and theeffect of the clock buffers can be neglected for choosing .The maximum number of is limited to the target clock frequency. As the minimum clock cycle timeis , where is thedelay from the rising edge of the main clock signal (CLK) tothe rising edge of the first pulsed clock signal (CLK_pulse(T)),is the delay of two neighbor pulsed clock signals,is the delay from the rising edge of the last pulsed clock signal(CLK_pulse(1)) to the output signal of the latch Q1 is proportional to . As increases, the maximum clock frequencydecreases in proportion to. Therefore, must be selected under the maximumnumber which is determined by the maximum clock frequencyof the target applications.The pulsed clock signals are supplied toall sub shift registers. Each pulsed clock signal arrives at thesub shift registers at different time due to the pulse skew in thewire. The pulse skew increases proportional to the wire distancefrom the delayed pulsed clock generator. All pulsed clock signalshave almost the same pulse skews when they arrive at thesame sub shift register. Therefore, in the same sub shift register,the pulse skew

differences between the pulsed clock signals arevery small. The clock pulse intervals larger than the pulse skewdifferences cancel out the effects of the pulse skew differences.Also, the pulse skew differences between the different sub shiftregisters do not cause any timing problem, because two latchesconnecting two sub shift registers use the first and last pulsedclocks (CLK_pulse(T) and CLK_pulse(1)) which have a longclock pulse interval.

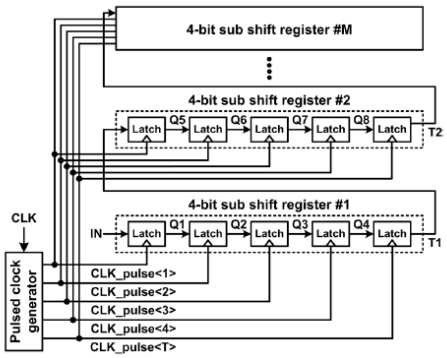


Fig3.1: Pulsed latch architecture

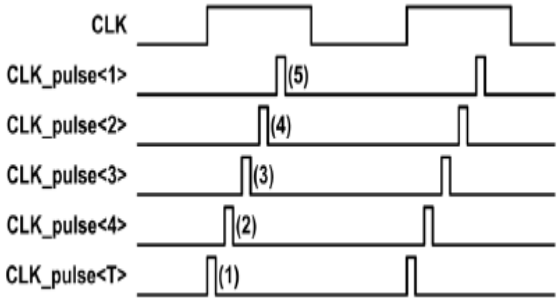


Fig3.2 : Clock Signals

3.1 Delayed clock pulse generator:

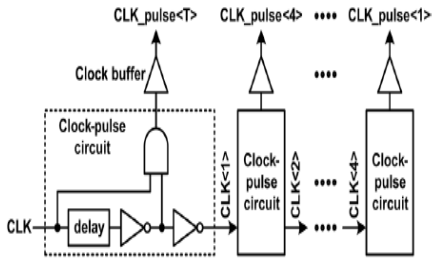


Fig3.3 : Delayed Clock Pulse Generator

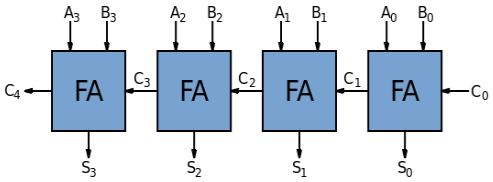
In a long shift register, a short clock pulse cannot through a long wire due to parasitic capacitance and resistance. At theend of the wire, the clock pulse shape is degraded because the rising and falling times of the clock pulse increase due to thewire delay. A simple solution is to increase the clock pulsewidth for keeping the clock pulse shape. But this decreases themaximum clock frequency. Another solution is to insert clockbuffers and clock trees to send the short clock pulse with a smallwire delay. But this increases the area and power overhead.

Moreover, the multiple clock pulses make the more overheadfor multiple clock buffers and clock trees.

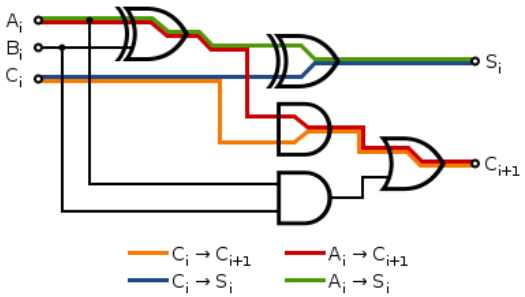
3.2. Ripple carry adder

A ripple carry adder is simply several full adders connected in a series so that the carry must propagate through every full adder before the addition is complete. Ripple carry adders require the least amount of hardware of all adders, but they are the slowest.

The following diagram shows a four-bit adder, which adds the numbers A[3:0] and B[3:0], as well as a carry input, together to produce S[3:0] and the carry output.



3.4 Propagation Delay in Full Adders



Real logic gates do not react instantaneously to the inputs, and therefore digital circuits have a maximum speed. Usually, the delay through a digital circuit is measured in gate-delays, as this allows the delay of a design to be calculated for different devices. AND and OR gates have a nominal delay of 1 gate-delay, and XOR gates have a delay of 2, because they are really made up of a combination of ANDs and ORs.

A full adder block has the following worst case propagation delays:

- From A_i or B_i to C_{i+1}: 4 gate-delays (XOR → AND → OR)
- From A_i or B_i to S_i: 4 gate-delays (XOR → XOR)
- From C_i to C_{i+1}: 2 gate-delays (AND → OR)
- From C_i to S_i: 2 gate-delays (XOR)

Because the carry-out of one stage is the next's input, the worst case propagation delay is then:

- 4 gate-delays from generating the first carry signal ($A_0/B_0 \rightarrow C_1$).
- 2 gate-delays per intermediate stage ($C_i \rightarrow C_{i+1}$).
- 2 gate-delays at the last stage to produce both the sum and carry-out outputs ($C_{n-1} \rightarrow C_n$ and S_{n-1}).

So for an *n*-bit adder, we have a total propagation delay, t_p of:

This is linear in *n*, and for a 32-bit number, would take 66 cycles to complete the calculation. This is rather slow, and restricts the word length in our device somewhat. We would like to find ways to speed it up.

IV. KOGGE-STONE ADDER

The Kogge–Stone adder is a parallel prefix form carry look-ahead adder. Other parallel prefix adders include the Brent-Kung adder, the Han Carlson adder, and the fastest known variation, the Lynch-Swartzlander Spanning Tree adder.

The Kogge–Stone adder takes more area to implement than the Brent–Kung adder, but has a lower fan-out at each stage, which increases performance for typical CMOS process nodes. However, wiring congestion is often a problem for Kogge–Stone adders. The Lynch-Swartzlander design is smaller, has lower fan-out, and does not suffer from wiring congestion; however to be used the process node must support Manchester Carry Chain implementations. The general problem of optimizing parallel prefix adders is identical to the variable block size, multi level, carry-skip adder optimization problem, a solution of which is found in.

An example of a 4-bit Kogge–Stone adder is shown in the diagram. Each vertical stage produces a "propagate" and a "generate" bit, as shown. The culminating generate bits (the carries) are produced in the last stage (vertically), and these bits are XOR'd with the initial propagate after the input (the red boxes) to produce the sum bits. E.g., the first (least-significant) sum bit is calculated by XOR'ing the propagate in the farthest-right red box (a "1") with the carry-in (a "0"), producing a "1". The second bit is calculated by XOR'ing the propagate in second box from the right (a "0") with C0 (a "0"), producing a "0".

Enhancements to the original implementation include increasing the radix and sparsity of the adder. The *radix* of the adder refers to how many results from the previous level of computation are used to generate the next one. The original implementation uses radix-2, although it's possible to create radix-4 and higher. Doing so increases the power and delay of each stage, but reduces the number of required stages. The *sparsity* of the adder refers to how many carry bits are generated by the carry-tree. Generating every carry bit is called sparsity-1, whereas generating every other is sparsity-2 and every fourth is sparsity-4. The resulting carries are then used as the carry-in inputs for much shorter ripple carry adders or some other adder design, which generates the final sum bits. Increasing sparsity reduces the total needed computation and can reduce the amount of routing congestion.

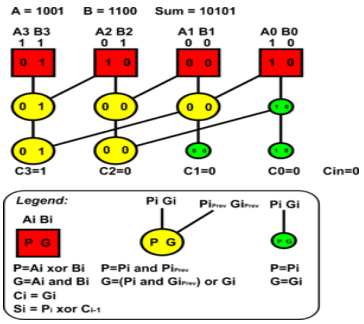


Fig:4.1 4-bit Kogge–Stone adder with zero carry-in.

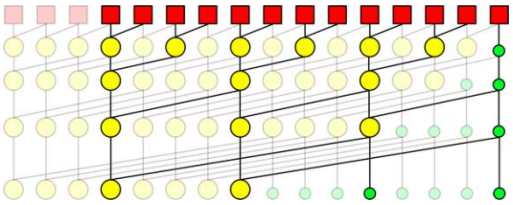


Fig:4.2 implementation of kogge stone adder

V. RESULTS

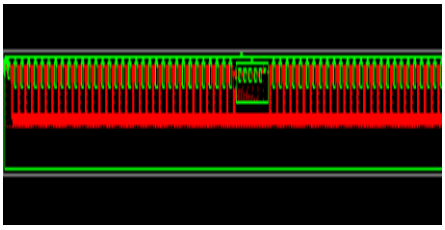


Fig : RTL Schematic

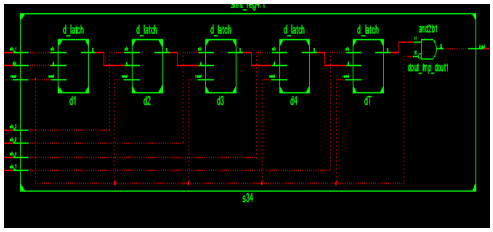


Fig : RTL internal structure

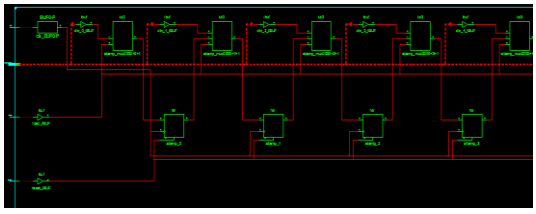


Fig : Technology Schematic

VI. SIMULATION RESULT

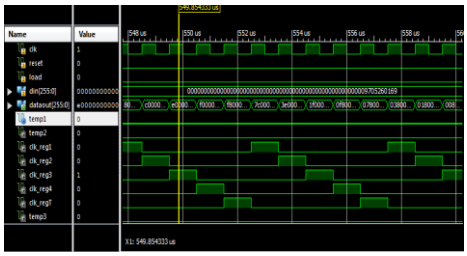


Fig : simulation results and related timing diagram

VII. CONCLUSION

This project proposes delay efficient shift register by using pulsed latches. Pulsed latches gives the same results similar to the flip flops but with less hardware resources. Flip flops need two latches. The same results can be obtained by using a clocked latch with pulse signal as clock input. This reduces the hardware requirement, in addition to this for producing different clock pulses for latches this project uses counter architecture for activating the latches. In this, counter is the circuit that increments its previous value by 1. In the increasing process several adders are implemented. This project proposes kogge stone adder which takes 0.932 nano seconds where as the conventional adder takes 3.497 nano seconds which reduces ¾ of the conventional adder delay.

VIII. REFERENCES

[1] P. Reyes, P. Reviriego, J. A. Maestro, and O. Ruano, "New protectiontechniques against SEUs for moving average filters in a radiation environment,"IEEE Trans. Nucl. Sci., vol. 54, no. 4, pp. 957–964, Aug.2007.

[2] M. Hatamianet al., "Design considerations for gigabit ethernet 1000base-T twisted pair transceivers," Proc. IEEE Custom Integr. CircuitsConf., pp. 335–342, 1998.

[3] H. Yamasaki and T. Shibata, "A real-time image-feature-extraction andvector-generation vlsi employing arrayed-shift-register architecture,"IEEE J. Solid-State Circuits, vol. 42, no. 9, pp. 2046–2053, Sep. 2007.

[4] H.-S. Kim, J.-H. Yang, S.-H. Park, S.-T. Ryu, and G.-H. Cho, "A 10-bitcolumn-driver IC with parasitic-insensitive iterative charge-sharingbased capacitor-string interpolation for mobile active-matrix LCDs,"IEEE J. Solid-State Circuits, vol. 49, no. 3, pp. 766–782, Mar. 2014.

[5] S.-H. W. Chiang and S. Kleinfelder, "Scaling and design of a 16-megapixelCMOS image sensor for electron microscopy," in Proc. IEEENucl. Sci. Symp. Conf. Record (NSS/MIC), 2009, pp. 1249–1256.

[6] S. Heo, R. Krashinsky, and K. Asanovic, "Activity-sensitive flip-flopand latch selection for reduced energy," IEEE Trans. Very Large ScaleInteger. (VLSI) Syst., vol. 15, no. 9, pp. 1060–1064, Sep. 2007.

[7] S. Naffziger and G. Hammond, "The implementation of the nextgeneration64 b itanium microprocessor," in IEEE Int. Solid-State CircuitsConf. (ISSCC) Dig. Tech. Papers, Feb. 2002, pp. 276–504.

[8] H. Partoviet al., "Flow-through latch and edge-triggered flip-flop hybridelements," IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech.Papers, pp. 138–139, Feb. 1996.

[9] E. Consoli, M. Alioto, G. Palumbo, and J. Rabaey, "Conditionalpush-pull pulsed latch with 726 fJops energy delay product in 65 nmCMOS," in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech.Papers, Feb. 2012, pp. 482–483.

[10] V. Stojanovic and V. Oklobdzija, "Comparative analysis of masterslavelatches and flip-flops for high-performance and low-power systems,"IEEE J. Solid-State Circuits, vol. 34, no. 4, pp. 536–548, Apr.1999.

[11] J. Montanaroet al., "A 160-MHz, 32-b, 0.5-W CMOS RISC microprocessor,"IEEE J. Solid-State Circuits, vol. 31, no. 11, pp. 1703–1714,Nov. 1996.

[12] S. Nomura et al., "A 9.7 mW AAC-decoding, 620 mW H.264 720p60fps decoding, 8-core media processor with embedded forwardbody-biasing and power-gating circuit in 65 nm CMOS technology,"in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers,Feb. 2008, pp. 262–264.

[13] Y. Ueda et al., "6.33 mW MPEG audio decoding on a multimedia processor,"in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers,Feb. 2006, pp. 1636–1637.

[14] B.-S. Kong, S.-S. Kim, and Y.-H. Jun, "Conditional-capture flip-flopfor statistical power reduction," IEEE J. Solid-State Circuits, vol. 36,pp. 1263–1271, Aug. 2001.

[15] C. K. Teh, T. Fujita, H. Hara, and M. Hamada, "A 77% energy-saving22-transistor single-phase-clocking D-flip-flop with adaptive-couplingconfiguration in 40 nm CMOS," in IEEE Int. Solid-State Circuits Conf.(ISSCC) Dig. Tech. Papers, Feb. 2011, pp. 338–339.

