Table of Contents Search Author Index

Hot Block Ring Counter: A Low Power Synchronous Ring Counter

Mohammad Dastjerdi-Mottaghi
Nanoelectronics Center of Excellence, School of Electrical and Computer Engineering
University of Tehran, Tehran, Iran
Email: m.mottaghi@ece.ut.ac.ir

Anahita Naghilou
Nanoelectronics Center of Excellence, School of Electrical and Computer Engineering
University of Tehran, Tehran, Iran
Email: a.naghilou@ece.ut.ac.ir

Masoud Daneshtalab
Nanoelectronics Center of Excellence, School of Electrical and Computer Engineering
University of Tehran, Tehran, Iran
Email: daneshtalab@ece.ut.ac.ir

Ali Afzali-Kusha
Nanoelectronics Center of Excellence, School of Electrical and Computer Engineering
University of Tehran, Tehran, Iran

Email: afzali@ut.ac.ir

Zainalabedin Navabi
Nanoelectronics Center of Excellence, School of Electrical and Computer Engineering
University of Tehran, Tehran, Iran
Department of Electrical and Computer Engineering
Northeastern University, Boston, U.S.A.
Email: navabi@ece.neu.edu

***Abstract*— In this paper, we propose a new and low-power architecture for synchronous ring counters which can noticeably reduce the switching activity of the conventional ring counters. To achieve the goal we partition the ring counter into some blocks for each of which we use a special clock gator. The Hot block (the block in which the '1' exists) is the only block the flip-flops of which are clocked. The delay and area overhead of the proposed clock gator is independent of the block size; this enables designer to freely resize the blocks and compromise with area and power overheads. The latency increase in the proposed architecture is independent of the counter width and depends only on the technology. For 90 nm technology it increases the latency by5%. The architecture noticeably (about 85%) reduces the total switching activity of the counter especially for wide counters.**

Index Terms— **Cell driven, Clock gator, Entrance, Exit, Hot Block, Latch, Low-power, Ring Counter, Switching activity, Watchdog.**

I. INTRODUCTION

The power consumption of digital circuits has become a critically important parameter motivating many efforts in reducing the power dissipation of the logic blocks of digital systems. Among different blocks, ring counter is one of logic components which has several applications including control units [1] and [2], multiplier and divider architectures [3], and the arbitration circuitry (round robin arbitration) of routers [4]. One of the important properties of a ring counter is that its output is one-hot encoded (*i.e.*, there is always only a single '1'-valued bit in its output and all other bits are zero). This property of the ring counter makes its output wide especially as the counter size increases. As an example, consider a 5-bit binary counter which counts from 0 to 31. A ring counter with the same counting range is 32-bits wide.

In this paper we propose a new architecture (called Hot-Block) which lowers the switching activity of ring counters. The paper is organized as follows. In Section II, we present a simple clock gating scheme for ring counters. Then in section III the proposed architecture is explained. Simulation results are given in section IV and are discussed. Finally you will find summary and conclusion in section V.

II. CONVENTIONAL RING COUNTERS

The first step toward a low power design is to detect signals which can be temporarily or locally shot off without affecting the circuit functionality. Therefore we first inspect the logic of a ring counter.

A. Unnecessary Transitions in Conventional Ring Counters

An *n*-bit synchronous ring counter is built up by cascading *n* D-flip-flops in a chain as depicted in "**Fig. 1**".

1-4244-0765-6/06/$20.00 ©2006 IEEE

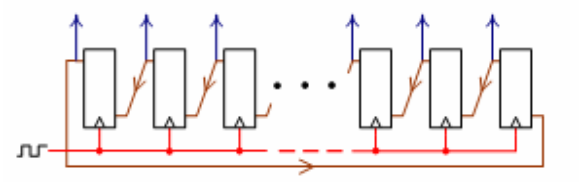

Fig. 1. The conventional synchronous ring counter [5]

In this architecture all flip-flops have a common clock signal and each clock pulse is applied to all flip-flops whereas inspecting the movement of the '1' in the counter chain reveals that each clock pulse must be applied to only 2 flip-flops (not all of them). Therefore on each clock pulse $(n–2)\times s$ unnecessary transitions are raised in which s is the total number of transitions raised in a single flip-flop.

B. Cell Driven Clock Gating Scheme

According to the previous discussion some flip-flops can be clock-gated, leading to fewer switching activities. *A flip-flop in a ring counter must be clocked if and only if either its input or its output is '1' immediately before the triggering clock edge comes*. Therefore only 2 flip-flops must be clocked in each cycle (as expected).

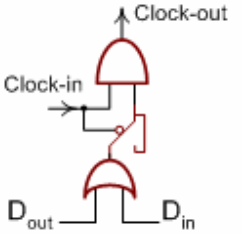

Fig. 2. The single bit clock gator logic. – D_{in} and D_{out} are flip-flop's input and output respectively.

The cited rule helps us design a clock gating logic which is shown in "**Fig. 2**". As seen in the figure, it ORs the values of flip-flop's input and output and on the positive clock edge, stores the result in a latch. The output of the latch determines whether or not to gate the clock signal. This clock gator is positive edge triggered; i.e. it is designed for positive edge triggered flip-flops.

If we want to avoid all unnecessary transitions raised by the clock signal, we should provide each flip-flop with the clock gating circuitry of "**Fig. 2**"; but unfortunately this solution ends up with a large area overhead plus due to transitions in clock gators themselves, the resulting ring counter will not have fewer switching activity.

A better solution is to use a single clock gator for multiple flip-flops. In this case the counter should be partitioned to b blocks of f flip-flops as shown in "**Fig. 3**". In this figure f is 3.

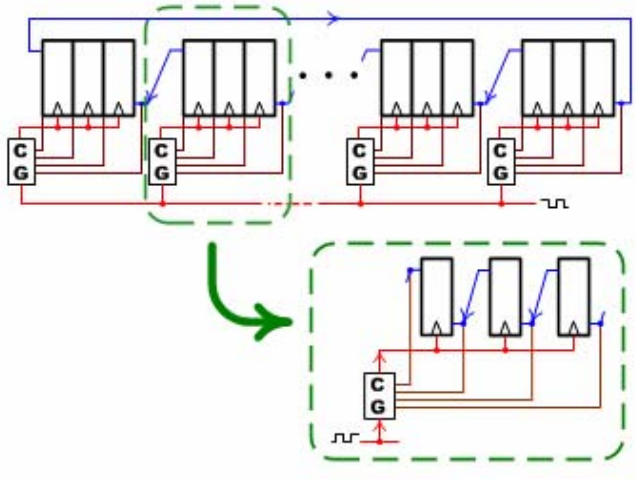

Fig. 3. Using a single clock gator for multiple (3 here) flip-flops

The clock gator in this architecture is similar to that of "**Fig. 2**" except the OR gate which should be widened to accommodate more flip-flop outputs. In "**Fig. 4**" a multiple bit clock gator (used in the architecture of "**Fig. 3**") is shown.

In the architecture of "**Fig. 3**" there is always at least one clock gator which is open, hence there is always one OR gate the inputs of which are altering. In fact in this architecture signal transition is avoided in the blocks but we have a new source of switching activity (absent in the conventional architecture) in the inputs of the clock gators: the OR gates are still active.

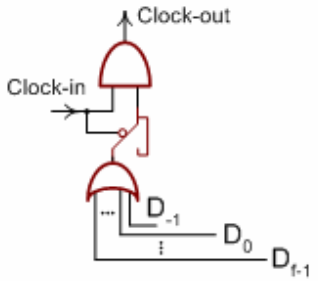

Fig. 4. The multiple bit clock gator logic. – For a block with f flip-flops, the OR gate has $f+1$ inputs. - D_{-1} is connected to the leftmost bit of the righthand block.

Although reduces the total switching activity of the ring counter, the multiple bit clock gator of "**Fig. 4**" is problematic in terms of area overhead and switching activity. As discussed before the OR gate of the logic widens as the block becomes wider and poses fan-in problems; as an example for 8-bit blocks the OR gate has 9 inputs which should be divided to 3 OR gates with fewer inputs.

III. "HOT BLOCK" ARCHITECTURE

It would be very nice if the OR gate (in **Fig. 4**) could be prevented from widening. Specifically the clock gator itself should not be a source of unnecessary switching activity. We have designed a multiple bit clock gator, the power and

area overhead of which is constant and independent of the size of the block it clock-gates. Plus it raises no unnecessary signal transition.

In order to have a low activity ring counter we propose to partition the ring counter into *b* blocks each of which is clock-gated with a special clock gator as shown in "**Fig. 5**".

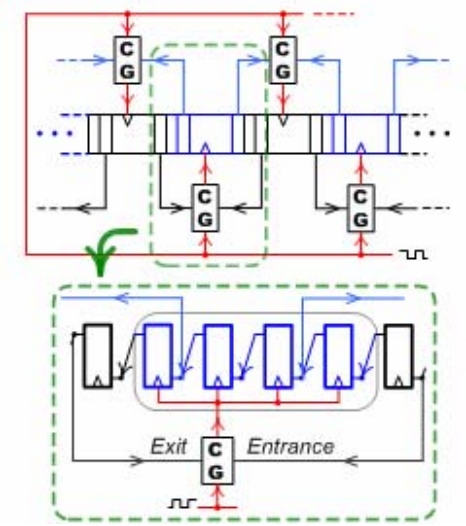

Fig. 5. Hot Block architecture for ring counters. – The ring counter is partitioned into *b* blocks of size *s* (*s* is 4 in this figure). – The complexity of the clock gators remains unchanged with varying block sizes.

Comparing the architecture of "**Fig. 3**" with that of "**Fig. 5**" reveals the elegance of the proposed architecture: in the architecture of "**Fig. 3**" the output of each flip-flop goes to one of clock gators (hence called *cell driven* scheme; see the inputs of the clock gators, see also **Fig. 4**) whereas in the *Hot Block* (proposed) architecture there are many flip-flops the output of which does not go to any clock gator. This point noticeably reduces the total switching activity of the ring counter.

A ring counter has a very interesting property of which we have taken advantage in design of our architecture: *a '1' is moving through the cells of the ring counter, and at each moment all cells of the ring counter except one is zero(0).* This property helps us remove the OR gate from the clock gator of "**Fig. 4**". The cited property tells us that in the partitioned ring counter of "**Fig. 5**" there is always only one block the flip-flops of which should be clocked (except in some occasions where the '1' leaves a block and enters another); this block is called the *Hot Block* (hence the architecture name). Therefore for each block the clock gator should only know whether the '1' has entered that block and also has not left it yet. In other words the '1' moves through the counter from right to left. During its movement, it enters a block from right and after a while (in a matter of some clock pulses) leaves that block from left and enters another block (the left hand neighbor of the previous block). So the clock gator of a block should wait until the '1' enters that block; once it is entered, the clock gator starts passing all clock pulses to the flip-flops of the block. It keeps passing clock pulses until the '1' leaves the block in case of which, shuts off clock pulses.

As described in the previous paragraph it is not necessary for the clock gator to OR all flip-flop outputs of the block; since we are sure that there is only one flip-flop with the value of '1'. Therefore it is sufficient to watchdog the entrance and exit ports of the block. If the '1' enters the block we are certain that as long as it has not left the block, all clock pulses should be passed to the block cells. As soon as the '1' leaves the block, no clock pulse should be passed to the block cells. As you see, taking the advantage of this nice property we can remove the OR gate from the clock gator of "**Fig. 4**" and devise a new clock gator which is shown in "**Fig. 6**".

A. How The Clock Gator Works

The proposed clock gator, shown in "**Fig. 6**", is composed of four multiplexers and a latch. In addition to *reset* and *clock-in* signals there are two other signals coming from neighboring blocks (left and right) named *Entrance* and *Exit*. These two signals are used to determine whether or not the '1' is present in the block to which the clock gator output goes.

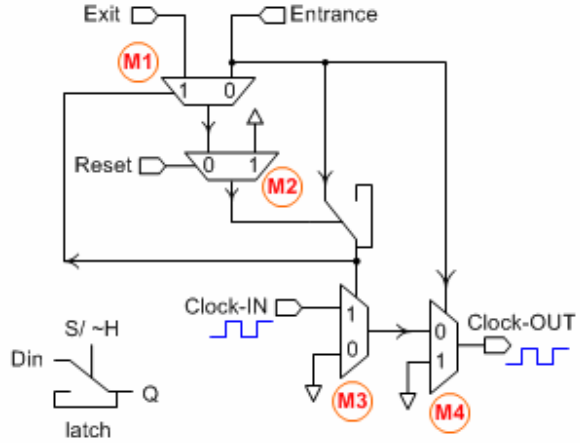

Fig. 6. The Hot Block clock gator used in the Hot Block architecture (Fig. 5) – Whatever the block size is, there are only two inputs (*Entrance* and *Exit*) coming from other blocks.

The active-high *reset* signal is used to reset the clock gator. When '1', the *reset* signal sets the *sample* line *(S/~H)* of the latch to '1', through multiplexer *M2*. This causes the latch to read the *Entrance* signal which is already reset to '0' (since the whole ring counter is reset.). After a sufficiently long interval, *reset* goes to '0' which causes the value of the *Entrance* signal to be placed on the *~hold* line of the latch, through multiplexers *M1* and *M2*. Since *Entrance* has the value of '0', the latch holds zero on its output after the clock gator is reset.

Multiplexer *M1* plays the *watchdog* role about which we talked in the previous section. After the clock gator is reset the selector line of multiplexer *M1*, has the value of zero which causes the *Entrance* signal to be selected (watch dogged) by this multiplexer. As seen in the figure, the output of the latch is also connected to the selector line of multiplexer *M3* which causes the input clock signal to be shut off (gated), after the clock gator is reset.

Top

The *Entrance* and *Exit* signals have special meanings; when one, *Entrance* means that *the '1' is about to enter the block in the next cycle.* This line is connected to the **input** of the leftmost flip-flop of the right hand neighbor of the block (see "**Fig. 5**"). Therefore whenever it is '1' we can deduce that the '1' is about to leave the right hand neighbor and enter the block (the '1' is moving from right to left). The *Exit* signal on the other hand signals that *the '1' has left the block and it shall not be clocked.*

Once the *Entrance* signal becomes '1', the *sample* and *data-in* lines of the latch are set to '1' which causes multiplexer *M1*, to select (watch dog) the *Exit* signal which is already zero (since all cells of the ring counter except one, have the value of zero in them.). Through multiplexers *M1* and *M2*, the value of the *Exit* signal (0) goes to the *~hold* line of the latch which causes the latch to hold '1' (the value of the *Entrance* signal) on its output. From this moment on, the *Exit* signal is watch dogged by multiplexer *M1*, plus clock pulses are no longer gated by multiplexer *M3*. Note that although no clock pulse is gated, but in the cycle when *Entrance* just becomes '1', the clock pulse is masked by means of multiplexer *M4* (see "**Fig. 6**"). This is because in this cycle the block cells shall not be clocked and the clock gator should only prepare to pass the input clock pulses in the **next** cycles.

Clock pulses come to the clock gator and propagate through multiplexers *M3* and *M4* and go to the block cells via *Clock-OUT*, until the *Exit* signal becomes '1' in which case the *sample* line of the latch becomes '1' through multiplexers *M1* and *M2*. This causes the latch to read its input (the *Entrance* signal) which is zero at this time. Zero propagates through the latch and reaches the selector line of multiplexer *M1* causing the *Entrance* signal to be watch dogged again. The output of the latch (which is zero) in this state also makes multiplexer *M3* to shut off the input clock pulses.

To summarize, after the clock gator is reset the *Entrance* signal is watch dogged and all clock pulses are shut off. This continues until *Entrance* becomes '1' in which case all clock pulses are passed to the output and the *Exit* signal becomes watch dogged. Once *Exit* becomes '1', the *Entrance* signal becomes watch dogged again and the whole scenario repeats in this manner.

As you see, no mater how wide the block size is, the proposed clock gator ("**Fig. 6**") has a total of 4 inputs. This is in contrast to the clock gator of "**Fig. 4**" the input count of which increases as the block widens.

As an implementation point we should point out that the rightmost flip-flop is reset to '1' (not zero) and also its associated clock gator is reset to *open* (not shut off) state. Note also that *Entrance* of this clock gator comes from the input of the left-most flip-flop of the counter.

IV. Results and Discussion

We implemented both the conventional (**Fig. 1**) and *Hot Block* (**Fig. 5**) architectures for 16-, 32-, 48- and 64-bit ring counters. In the case of Hot Block ring counters we made three instances with block sizes of 4, 6 and 8 bits (flip-flops). The results are shown in "**Fig. 7**" and "**Fig. 8**". As seen in "**Fig. 7**" for 64-bit ring counter with blocks of 4 flip-flops we had 84.5% activity reduction. In "**Fig. 8**" you see how efficacious (and low power dissipating) the *Hot Block* architecture is for wide ring counters (e.g. 64-bit).

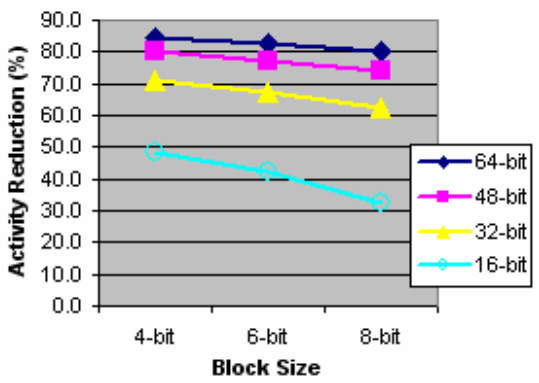

Fig. 7. The activity redutcions for ring counters of different width and block sizes as compared to the conventional ring counter

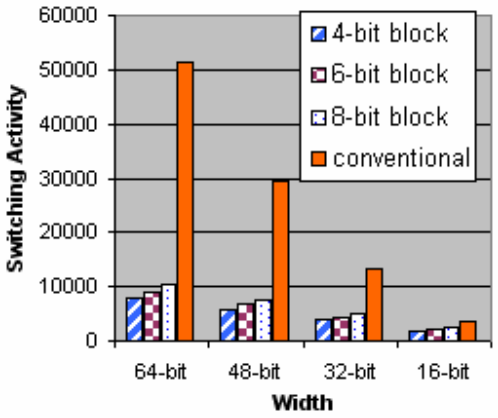

Fig. 8. Switching activities of different ring counters – Hot Block ring counters raise noticeably fewer signal transitions. – Best results are obtained in Hot Block ring counters with blocks of size 4.

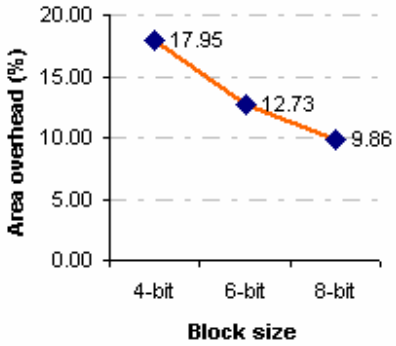

Fig. 9. The area overhead for different block sizes – as expected smaller blocks have larger area overheads

The Hot Block clock gator (**Fig. 6**) can be implemented using 14 transistors (6 for the latch and 8 for the multiplexers (each 2)). Each flip-flop takes 16 transistors.

Therefore for a block of size f (with f flip-flops) the area overhead of the Hot Block clock gator, in terms of the number of transistors is:

$$AreaOverhead = \frac{14}{14 + f \times 16}$$

"**Fig. 9**" shows the trend of area overhead for different block sizes; for example Hot Block ring counters with blocks of size 4 (4 flip-flops) require 17.9% more transistors than conventional ring counters.

Inspecting the Hot Block architecture and comparing "**Fig. 1**" with "**Fig. 5**" reveals that the critical path of the Hot Block architecture is the same as that of the conventional architecture except that the clock signal passes through a clock gator in the Hot Block architecture. A closer look at "**Fig. 6**" shows that the cited critical path passes through multiplexers *M3* and *M4*. More precisely, two source-to-drain channels are added to the critical path of the Hot Block architecture. The two channels are charged before the clock edge comes and the charging time does not affect the path delay. The capacitance overhead is negligible. Therefore R_{load} of the clock generator is added by $2 \times R_{channel}$. This augmentation in the resistance of the clock path results in a latency in clock pulses which is technology dependant and independent of the width of the counter; this is because the Hot Block clock gator (**Fig. 6**) remains unchanged with varying counter width. Simulation results for a 90 nm technology show that the latency increase is about 5%. The area overhead is dependant on the block size and for larger blocks we have less area overheads but the power consumption is more.

V. Summary and Conclusion

In this work we proposed the Hot Block architecture (**Fig. 5**) for ring counters which considerably lowers their total switching activity. The proposed architecture was based on partitioning the counter into blocks of flip-flops. A clock gating circuit (clock gator, **Fig. 6**) was used to allow the clock to be selectively applied to the blocks, causing only the block containing the "1", called the Hot Block, to be clocked and the clock to be masked for other blocks. The proposed clock gator had a very important and useful property that its complexity was independent of the size of the block it clock gates, giving chance to designer to freely resize the blocks to compromise the area overhead and activity (power) reduction.

Simulation results shows that in comparison with the conventional architecture, the Hot Block architecture reduces the switching activity more than 84% for 64-bit counters (see "**Fig. 7**" and "**Fig. 8**"). The proposed architecture for blocks of size 4, 6 and 8 has less than 18%, 12% and 10% area overhead respectively (see "**Fig. 9**").

The latency in the *Hot Block* architecture, which is caused by the clock gator, is constant and dependant on the technology (independent of the counter width). For 90 nm technology we had a latency increase of 5%.

REFERENCES

[1] Sarwate D.V. Shanbhag N.R., "High-speed architectures for Reed-Solomon decoders," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Volume 9, Issue 5, Oct. 2001 pp 641 – 655.

[2] Lin, Y.-T. Tsai, P.-Y. Chiueh, T.-D., "Low-power variable-length fast Fourier transform processor," in *IEE Proceedings on Computers and Digital Techniques*, Volume 152, Issue 4, July. 2005 pp 499 – 506.

[3] Mohammad D. Mottaghi, Ali Afzali-Kusha, Zainalabedin Navabi, "ByZFAD: A Low Switching Activity Architecture for Shift-and-Add Multipliers," accepted in SBCCI 2006, to be published.

[4] Cesar Albenes Zeferino1, Altamiro Amadeu Susin1, "SoCIN: a parametric and scalable network-on-chip," 16th Symposium on Integrated Circuits and Systems Design (SBCCI 2003), Sept. 2003, pp 169-174

[5] Victor P. Nelson, H. Troy Nagle, Bill D. Carroll J. David Irwin, Digital Logic Circuit Analysis & Design, Prentice-Hall, Inc., 1996

[6] Shams, A.M. Darwish, T.K. Bayoumi, M.A., "Performance analysis of low-power 1-bit CMOS full adder cells," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Volume 10, Issue 1, Feb. 2002 pp 20-29.