

# EFFICIENT TECHNIQUES BASED ON GATE TRIGGERING FOR DESIGNING STATIC CMOS ICs WITH VERY LOW GLITCH POWER DISSIPATION\*

NIHAR R. MAHAPATRA  
mahapatr@cse.buffalo.edu

Department of Computer Science & Engineering  
State University of New York at Buffalo  
Buffalo, NY 14260, USA

SRIRAM V. GARIMELLA  
garimels@cdg.stsv.seagate.com

VLSI Systems Group  
Seagate Technology, Inc.  
Scotts Valley, CA 95066, USA

ALWIN TAREEN  
tareen@eng.buffalo.edu

Department of Electrical Engineering  
State University of New York at Buffalo  
Buffalo, NY 14260, USA

## Abstract

*This paper presents a new framework called gate triggering for systematically minimizing glitch power dissipation in static CMOS ICs. It is based on the idea that glitches can be effectively minimized by triggering logic evaluation at a gate only when all of its inputs have stabilized. For this purpose, to every potentially glitchy gate (or a suitable subset of such gates) is added a small amount of control logic, which, when enabled, triggers logic evaluation at the gate. A clocked delay chain is employed to generate enable signals at the proper times for all gates to be triggered. We present six specific techniques based on gate triggering that differ in the type of control logic and the way it is used to control a gate. These techniques have varying effectiveness and area and timing overheads, which we analyze in detail. Application of these techniques to test circuits yields promising results.*

## 1 Introduction

Increasing levels of device integration, die size, and operating frequency, a burgeoning portable and embedded computing and communications market, combined with reliability and packaging cost concerns, have made power dissipation a major issue in VLSI design [10, 11]. In complementary static CMOS, a popular VLSI logic style, power is primarily dissipated during logic transitions when gate load capacitances charge and discharge.

While some logic transitions are necessary and are dictated by circuit functionality, others, such as glitches, are not. *Glitches* are spurious transitions that occur before a gate output reaches a stable value and are caused by unequal propagation delays of input signals to the gate (see Fig. 1(a)). Also, glitches multiply as they propagate through a combinational logic block (see Fig. 1(a)). Glitch power is typically significant and can be as high as 70% of total power dissipation in some cases [10]. As we go into deeper submicron technologies, interconnect delays become more predominant, which leads to differential delays and more glitching [13].

In this paper, we present a new framework called *gate triggering* for minimizing glitch power dissipation in complementary static CMOS ICs which we describe in the next section. An added advantage of our approach is that short-circuit power dissipation at gates that are controlled is also minimized. Next in Sec. 3, we describe and analyze six specific techniques based on gate triggering and also provide simulation results for them. Then in Sec. 4 we briefly discuss related previous work. Conclusions are in Sec. 5.

## 2 Proposed Methodology: Gate Triggering

The key idea we employ to minimize glitches is to trigger logic evaluation at a gate only after all of its inputs have stabilized. For this purpose, to every potentially glitchy gate (i.e., a gate with unequal propagation delays for its inputs) or a suitable subset of such

gates, we add some small control logic, which, when enabled, triggers logic evaluation at the gate (see Fig. 1(b)). Essentially, this logic controls gate connection to  $V_{DD}$  and/or  $V_{SS}$ ; various types of control logic are discussed in the next section. In order to enable the control logic for different gates in the combinational logic block of Fig. 1(b) at the proper times (i.e., when the last input to the individual gates has stabilized), we first perform a timing simulation of the combinational block.<sup>1</sup> From this, we obtain the delays of different gates and also the latest times by which the various inputs of a gate will have stabilized.

To prevent glitches at the output of a potentially glitchy gate, its control logic is enabled as soon as all inputs to the gate will have stabilized—the estimate of the time when all inputs become stable should be adjusted to account for the extra delay that the control logic may cause for gates to which it is added. An initial enable signal with a high period equal to the maximum delay for any gate in the block is generated from the clock signal (see Fig. 1(b)). This signal then passes through a chain of delay elements, whose outputs provide enable signals for potentially glitchy gates in the logic block. To minimize glitch power while keeping overheads (area, power dissipation, and increase in critical path delay due to extra logic, such as control logic and delay elements) low, only a subset of the potentially glitchy gates that provide maximal glitch power savings and affect critical path delay the least, should be controlled. A detailed analysis of overheads in our approach and an integer linear programming (ILP) formulation for minimizing these overheads is given in [6]. The delay element we used is a transmission gate or an inverter because these require relatively less area and/or consume very little power. A detailed discussion and comparison of delay elements motivating our choice is given in [7].

## 3 Glitch Minimization Techniques

### 3.1 Simulation Methodology

Here we present six specific techniques based on the gate triggering framework that differ in the type of control logic used and the way it is connected to a gate to prevent glitches. In order to more easily analyze and compare these techniques, we first apply them to a highly glitchy test gate. By studying the influence on one gate, the effectiveness of the techniques on a combinational logic block consisting of many gates can be inferred. The test gate and inputs used are shown in Fig. 2(a)—the asynchronous arrival of inputs to the test gate model unequal propagation delays from the input of a combinational block to the inputs of an interior gate. We analyze the test gate output for all four possible initial-final output value combinations: 0→0, 0→1, 1→0, and 1→1 (see Fig. 3). The original test gate (without glitch minimization) is referred to as *ckt0*. The test gate with one of the six techniques applied is referred to as one of *ckt1* through *ckt6*. Next, we discuss the six new glitch minimization techniques.

<sup>1</sup>Timing simulation is an essential step in the design flow of a VLSI chip [15] (e.g., to determine the critical-path delay in a combinational block, which in turn determines the clock period). Hence it does not represent an extra step in the application of our method.

\*N.R. Mahapatra was supported by startup funds from the State University of New York at Buffalo.

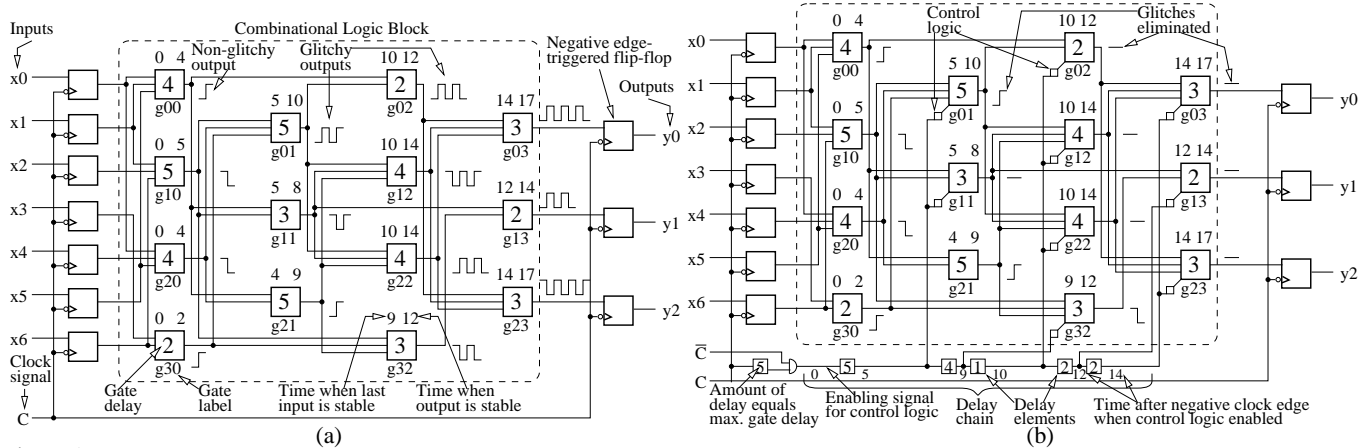


Figure 1: Minimizing glitch power dissipation in a synchronous sequential circuit design: (a) Glitches occurring in a combinational logic block at the outputs of gates that have multiple inputs changing asynchronously. (b) Glitches minimized by adding control logic to every potentially glitchy gate and enabling it through a delay chain when the last input to the gate stabilizes.

### 3.2 Technique 1

In this technique, the control logic is a transmission gate (T-gate), which is placed at the output of a potentially glitchy gate (Fig. 2(b)). The T-gate prevents glitch propagation to fanout gates by providing a reasonably clean output at  $F$  as seen in the second-from-left column of plots for ckt1 in Fig. 3. As seen in Fig. 1(a), glitches occurring in the earlier gates of a combinational logic block cause many more glitches at later gates. Thus, this technique can lead to significant glitch power reduction in combinational circuits. However, the technique has a number of shortcomings discussed next.

First, glitchy transitions still occur and power is dissipated at point  $F^*$  in Fig. 2(b). Second, a T-gate is needed for every potentially glitchy gate not connected to an output flip-flop. Third, introduction of a T-gate increases the overall delay if the gate is on a critical path of the combinational block. The delay overhead can be reduced by increasing the width of the transistors of the T-gate, and thereby reducing its resistance. Finally, to enable the T-gate, two complementary enable signals are needed, which necessitates two delay chains, one for generating the enable signal for the  $n$  transistor of the T-gate, and the other for its  $p$  transistor. Alternatively, an inverter may be used each time an enable signal is needed to generate the complementary enable signal, but care has to be taken to account for the delay of the inverter, which will make the two signals out of sync. Because of this and also because of the area required for an inverter, two delay chains may be preferable in most cases.

### 3.3 Technique 2

In contrast to the technique discussed above, the remaining techniques not only prevent glitch propagation from a gate, but also minimize glitch power dissipation at the gate itself. They do so by controlling the connection of the gate output to  $V_{DD}$  and/or  $V_{SS}$  by means of  $n$  and/or  $p$  control transistors. For instance, in the second technique (ckt2 in Fig. 2(c)), an  $n$  (or  $p$ ) control transistor is connected between the  $N$  pull-down ( $P$  pull-up) network of a potentially glitchy gate and  $V_{SS}$  ( $V_{DD}$ ). As a result, before the last input stabilizes, the output  $F$  of the gate can only rise to  $V_{DD}$ , since the path to  $V_{DD}$  is unobstructed, but it can not discharge to  $V_{SS}$ , since the  $n$  control transistor is off (see third column of plots in Fig. 3).

Therefore, a glitch (unnecessary transition) occurs with this technique only in one out of four cases when the both the initial and final values of the output are 0 (topmost plot in third column in Fig. 3). However, it should be noted that the output is partially glitchy in all cases in that, between the first and last input changes, it falls to some

appreciable extent and then rises. The reason for this is the following. In the previous clock cycle when the  $n$  control transistor was enabled, some of the  $n$  transistors in the  $N$  pull-down network were connected to  $V_{SS}$ , so that their drain capacitances were discharged to  $V_{SS}$ . In the current cycle, before the  $n$  control transistor is enabled, after the output rises to  $V_{DD}$ , these discharged internal capacitances may get connected to the gate output node because of some new input. This causes charge sharing between the output load capacitance and the internal capacitances of the  $N$  pull-down, thus bringing down the output voltage. These partial glitches consume some power and also get propagated, although to a lesser extent.

### 3.4 Technique 3

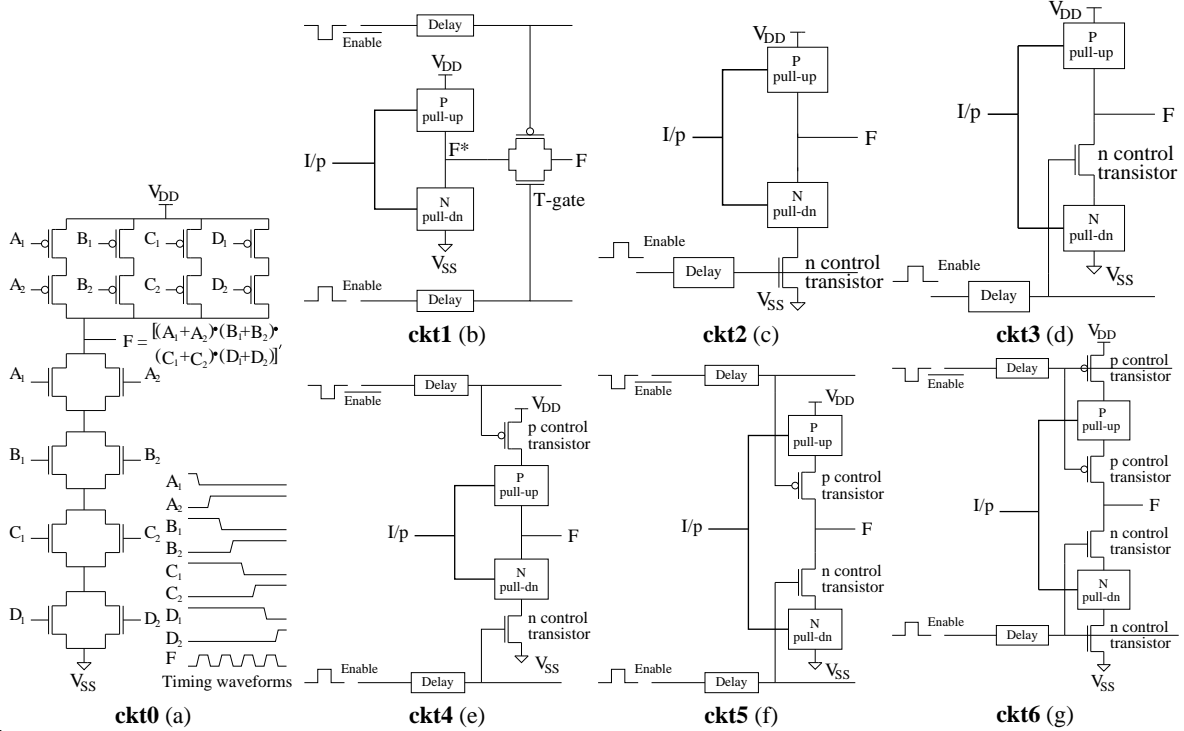
To prevent charge sharing and thus provide a cleaner output, in the next technique (ckt3 in Fig. 2(d)), an  $n$  (or  $p$ ) control transistor is connected instead between the output node and  $N$  pull-down ( $P$  pull-up). In this case too, the output can only rise to  $V_{DD}$  before the last input has stabilized, and there is a glitch only when the output has an initial and final value of 0 (fourth column of Fig. 3). One more peculiarity of this technique is that, when the final output value is high, the output temporarily dips a little before rising. This is because, when the  $n$  control transistor is enabled, some of the discharged internal capacitances of the  $N$  pull-down may get connected to the high output, bringing it down temporarily.

### 3.5 Technique 4

To prevent the gate output from charging to  $V_{DD}$  or discharging to  $V_{SS}$  before the last input has stabilized, and thus prevent the single glitch in both the previous techniques, the fourth technique (ckt4 in Fig. 2(e)) has an  $n$  control transistor between  $V_{SS}$  and  $N$  pull-down and a  $p$  control transistor between  $V_{DD}$  and  $P$  pull-up. Although, the output does not charge up or discharge completely, it does so partially when it has an initial high value (fifth column, bottom two plots in Fig. 3). This is because the initially charged up internal capacitances in the  $P$  pull-up and the output capacitance share charge with the initially discharged internal capacitances in the  $N$  pull-down. Thus when the initial-final output value is  $1 \rightarrow 0$  (fifth column, third plot from top in Fig. 3), the output discharges in step(s) or adiabatically, which actually means less power consumption than a straight discharge, and when the initial-final output value is  $1 \rightarrow 1$  (fifth column, bottom plot in Fig. 3), there is a partial glitch, which gets propagated to fanout gates.

### 3.6 Technique 5

To prevent charge sharing between the output capacitance and the internal capacitances of the  $N$  pull-down and  $P$  pull-up, and thus provide a cleaner, glitch-free output, the next technique uses an  $n$  control



**Figure 2:** Test circuit and various techniques to minimize glitch power dissipation: (a) Test circuit (ckt0) and input and output timing waveforms depicting a highly glitchy output. (b) A transmission gate placed at output of glitchy gate (ckt1) to filter glitches. (c) An n control transistor placed between N pull-down and  $V_{SS}$  (ckt2) prevents unnecessary discharging to  $V_{SS}$  of output capacitance and internal capacitances in N pull-down and P pull-up. (d) An n control transistor placed between N pull-down and output (ckt3) provides better output by preventing charge sharing between output capacitance and internal capacitances of N pull-down and P pull-up. (e) n and p control transistors connected to  $V_{SS}$  and  $V_{DD}$ , respectively, of output capacitance and internal capacitances in P pull-up and N pull-down. (f) n and p control transistors connected to output instead (ckt5) to provide better output by preventing charge sharing between output capacitance and internal capacitances of P pull-up and N pull-down. (g) n and p control transistors connected to  $V_{SS}$  and  $V_{DD}$  as well as to output (ckt6) to not only provide better output by preventing charge sharing between output capacitance and internal capacitances of P pull-up and N pull-down, but also to minimize power dissipation by preventing unnecessary charging to  $V_{DD}$  and discharging to  $V_{SS}$  of these capacitances.

transistor connected between N pull-down and output and a p control transistor between P pull-up and output (ckt5 in Fig. 2(f)). As seen in the sixth column of Fig. 3, the output is indeed much cleaner, except for the minor dip when the initial-final output value is  $1 \rightarrow 1$  (bottom plot), which occurs because of connection of discharged internal capacitances in the N pull-down and charged capacitances in the P pull-up.

### 3.7 Technique 6

Although the output is glitch-free in the above technique, internal capacitances in the N pull-down and P pull-up may unnecessarily discharge to  $V_{SS}$  and charge to  $V_{DD}$ , respectively, which results in power consumption. This is prevented in the final technique (ckt6 in Fig. 2(g)) in which two n and two p control transistors are used to prevent unnecessary charge sharing and charging/discharging of output capacitance and internal capacitances in the N pull down and P pull-up, and thus provide a clean output and minimize glitch power dissipation.

### 3.8 Comparison of Glitch Minimization Techniques

We now summarize and compare the various techniques in terms of their glitch minimization capability, area overhead, timing delay overhead, and ease of application. (1) *Glitch Minimization Capability:* Technique 1 provides only glitch filtering while the rest also provide glitch minimization at the gate to which they are applied. Glitch minimization improves from technique 1 through 6, except that technique 4 may provide better glitch minimization than technique 5, because in technique 4, when the initial-final output value is  $1 \rightarrow 0$ , the output discharges in stages (adiabatically), which means lower power consumption, and because the only glitch that occurs is a partial glitch. This is borne out by Table 1. (2) *Area Overhead:* The control transistor overhead of techniques 2 and 3 are one, that of techniques 1, 4, and

5 are two, and that of technique 6 is four. The control transistor overheads for techniques 2, 4, and 6 can be reduced by sharing the transistors connected to  $V_{SS}$  or  $V_{DD}$  with other potentially glitchy gates that need to be triggered at a similar time, i.e., the same control transistor can be used to connect multiple gates (that need to be triggered at a similar time) to  $V_{SS}$  or  $V_{DD}$ . In this case, the shared transistors may need to be sized up (larger width) to avoid increasing the critical path delay of the combinational block. Further overhead reduction can be achieved by selectively triggering only those gates that are expected to contribute most to glitch (or short-circuit) power dissipation. Techniques 2 and 3 require a single delay chain, whereas the others require two delay chains. As pointed out in Sec. 2, a detailed analysis of overheads (control logic, delay elements, and wiring) of our approach and how to minimize them using an ILP-based approach is given in [6]. (3) *Delay Overhead:* Technique 1 increases gate delay which can be reduced by increasing transistor widths. Control transistors used in the remaining techniques increase fall and/or rise time, which effect can be similarly minimized by increasing transistor widths if it adversely affects critical path delay. (4) *Ease of Application:* Given a gate layout, techniques 1, 2, and 4 can be applied with no modification, whereas the others can be applied with slight modification since n/p control transistors need to be introduced between the N pull-down and P pull-up.

From the above analysis, overall, technique 4 is the best in that it has close-to-optimal glitch minimization capability and low overhead. However, in any particular application, one or more techniques may be used for different gates. For instance, technique 2 may be used for simpler gates (since it uses only one control transistor), technique 4 may be used for medium complexity gates, and technique 6 for com-

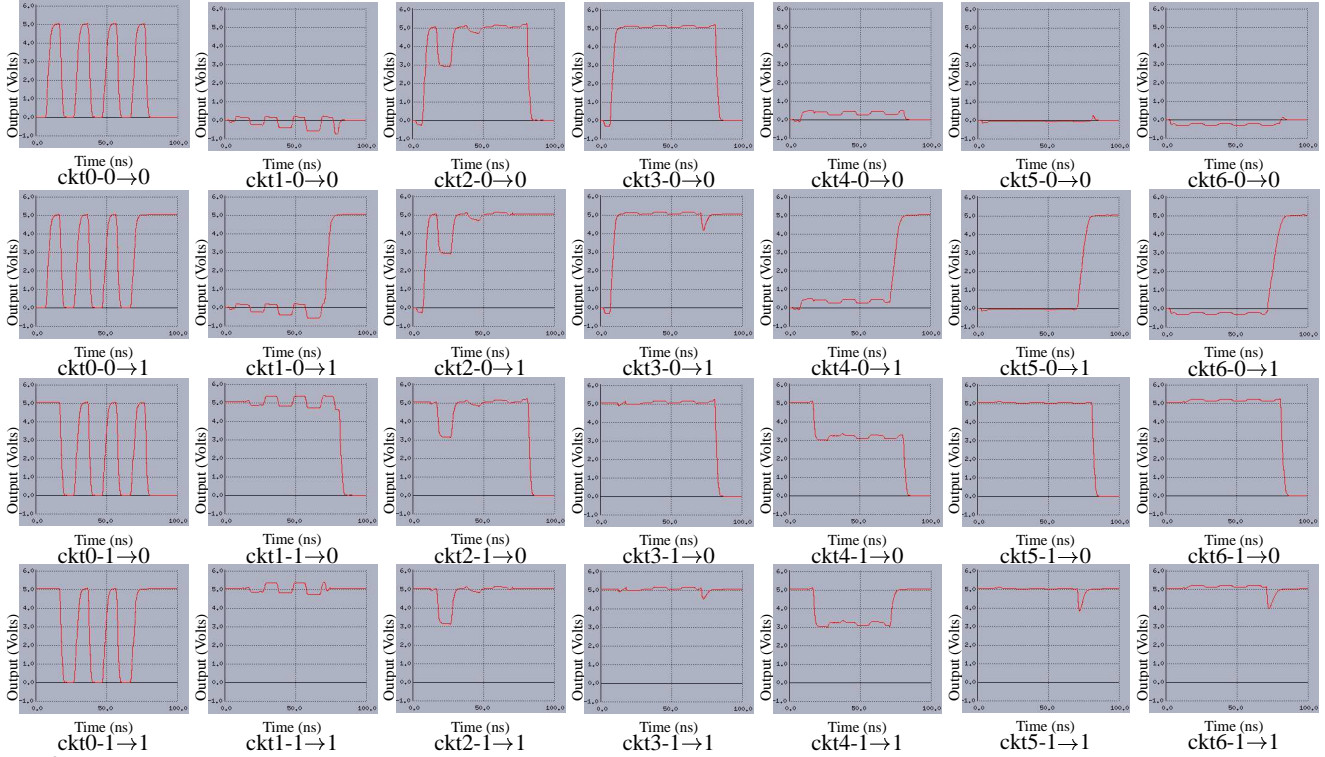


Figure 3: Output voltage waveforms for various glitch-minimization techniques for all four possible initial/final output value combinations. The naming format for plots is: < circuit technique > - < initial value of output → final value of output >. For all plots shown, fanout is four minimum-sized inverters and input inter-arrival time is two gate delays.

plex gates.

Circuit	Avg. Power ( $\mu W$ )	% Reduction	Circuit	Avg. Power ( $\mu W$ )	% Reduction
ckt0	98.56	—	ckt1	57.97	41.18
ckt2	36.01	63.46	ckt3	21.48	78.21
ckt4	5.42	94.50	ckt5	5.65	94.27
ckt6	3.62	96.33			

Table 1: Comparison of average power dissipation for the test circuit of Fig. 2(a) using various glitch-minimization techniques relative to the original circuit for input inter-arrival time equal to two gate delays and with a fanout of four inverters.

## 4 Related Work

Glitch and short-circuit power dissipation are discussed in [10, 11]. Glitch estimation, modeling, and propagation issues are covered in [2, 9, 13]. The importance of glitch minimization for various applications is considered in [3, 14, 16]. Designing two-level glitch-free circuits using logic redesign, assuming only one input changes at a time, is addressed in [4]. Glitch removal through path balancing obtained via, say, transistor sizing or layout changes, is discussed in [8, 10, 11]; this can be cumbersome and involves trial and error. Furthermore, in deep submicron technologies, transistor sizing will not be very effective for path balancing since logic delays become relatively smaller compared to interconnect delays. Retiming and buffer placement approaches to filter or reduce glitches and glitch propagation are described in [1, 5]; these approaches, although somewhat effective, entail appreciable area overheads for flipflops and buffers. Glitch reduction at the RT level in control flow intensive designs is given in [12].

Therefore, current methods for glitch reduction are either (i) not applicable in all contexts, or (ii) can not be automated and are ad hoc, or (iii) are not very effective, or (iv) have high area/delay overheads, or (v) restrict the manner in which logic is transformed to a gate realization. There is no methodical, generally applicable approach to minimizing glitch power dissipation. Our proposed gate triggering approach in this paper attempts to overcome all the above limitations of

current approaches.

## 5 Conclusions

Although research into various aspects of glitch power dissipation has been undertaken in the past, most approaches to addressing it are *ad hoc* and limited in their applicability. This paper presented a new framework called gate triggering for systematically minimizing glitch power dissipation in static CMOS ICs. Based on this framework, six specific techniques were proposed and their glitch minimization capabilities and overheads analyzed. Overall, technique 4 is the best in that it has close-to-optimal glitch minimization capability and low overhead. However, in any particular application, one or more techniques may be used for different gates, depending, for example, upon the gate's complexity. Application of the new approach to test circuits yields 95% or more elimination of glitch and, in gates to which applied, short-circuit power dissipation with very little area and timing overheads after optimization.

## References

- [1] M. Favalli and C. Metra, "The effect of glitches on CMOS buffer optimization," *Proc. PATMOS*, Oct. 1995 in pp. 202-212, Oldenberg, Germany, Oct. 1995.
- [2] M. Favalli and L. Benini, "Analysis of glitch power dissipation in CMOS IC's," *Proc. of ISLPED*, pp. 123-128, 1995.
- [3] "Gated clocks and hazards," <http://erebor.cudenver.edu/courses/ee3651/hazards/hazard.html>.
- [4] R.H. Katz, "Contemporary logic design," Addison Wesley Publications, 1993.
- [5] J. Leijten, et al., "Analysis and reduction of glitches in synchronous networks," *Proc. EDAC*, pp. 398-403, 1995.
- [6] N.R. Mahapatra and R. Janakiraman, "Gate triggering: A new framework for minimizing glitch power dissipation in static CMOS ICs and its ILP-based optimization," to be published in *Proc. IEEE ICCDCS 2000*, Cancun, Mexico, Mar. 15-17, 2000.
- [7] N.R. Mahapatra, S.V. Garimella, and A. Tareen, "An empirical and analytical comparison of delay elements and a new delay element design," to be published in *IEEE WVLST 2000*, Orlando, FL, Apr. 27-28, 2000.
- [8] E.J. McCluskey, "Logic design principles with emphasis on testable semicustom circuits," Prentice Hall, Englewood Cliffs, NJ, 1986.
- [9] F.N. Najm, "A survey of power estimation techniques in VLSI circuits," *IEEE T-VLSI*, Vol. 2, No. 4, Dec. 1994.
- [10] M. Pedram, "Power minimization in IC design: Principles and applications," *ACM TODAES*, Vol. 1, No. 1, pp. 3-56, Jan. 1996.
- [11] J.M. Rabaey, "Digital Integrate Circuits," Prentice Hall, 1996.
- [12] A. Raghunathan, S. Dey, and N.K. Jha, "Glitch analysis and reduction in register transfer level power optimization," *DAC*, pp. 331-336, Jun. '96.
- [13] S.J. Abou-Samra and A. Guyot, "Glitch threshold," *FTFC* 97.
- [14] P.S.K. Siegel, "Automatic technology mapping for asynchronous designs," Tech. Rep. CSL-TR-95-663, Stanford Univ.
- [15] N.H.E. Weste and K. Eshraghian, "Principles of CMOS VLSI design," Addison-Wesley, 1993.
- [16] T.-Y. Wu, et al., "A low glitch 10-bit 75-MHz CMOS video D/A converter," *IEEE JSSC*, Vol. 30, No. 1, Jan. 1995.