Digital Circuit Design for Minimum Transient Energy and a Linear Programming Method

Vishwani D. Agrawal
Bell Labs, Lucent Technologies
Murray Hill, NJ 07974 USA
va@research.bell-labs.com

Michael L. Bushnell
Rutgers University, CAIP Center
Piscataway, NJ 08855 USA
bushnell@caip.rutgers.edu

G. Parthasarathy
Rutgers University, CAIP Center
Piscataway, NJ 08855 USA
gpartha@caip.rutgers.edu

*Rajesh Ramadoss**
Lucent Technologies
Allentown, PA 18103 USA
ramadoss@lucent.com

Abstract

This paper provides a theoretical basis for eliminating or reducing the energy consumption due to transients in a synchronous digital circuit. The transient energy is minimized when every gate has no more than one output transition per clock cycle. This condition is achieved for a gate when the gate delay equals or exceeds the maximum difference between path delays at gate inputs. In practice, path delays are adjusted either by increasing gate delays or by inserting delay buffers. The minimum transient energy design is obtained when no delay buffer is added. This design requires possible increases in gate delays to meet the minimum energy condition at all gates. However, the delay of the critical path may be increased. In an alternative design, where the critical path delay is not allowed to increase, delay buffers may have to be added. The theory in this paper allows trade-offs between minimum transient energy and critical path delay. We formulate the problem as a linear program to obtain the minimum transient energy design with the smallest number of delay buffers for a given overall delay of the circuit. An optimized four-bit ALU circuit is found to consume 53% peak and 73% average power compared to the original circuit.

1. Introduction

Recently, Agrawal [1] proposed a method for low power design by increasing inertial delays of selected gates to suppress hazards. In this paper, we study the relationship between the energy consumed by hazards and the signal delays in the circuit. As a result, two methods of energy reduction emerge. These are the well-known method of *balanced delays* [2, 3] and the method of increasing delay for *hazard filtering* [1].

*Formerly with Rutgers University, CAIP Center, Piscataway, NJ 08855 USA

2. Circuit Delays and Transient Energy

Consider a combinational circuit consisting of Boolean gates. Primary inputs and the outputs of gates are signals. All signals are binary and assume either 0 or 1 values. Any change in a signal is called an *event*. Events at primary inputs of the combinational circuit start a *transition interval*, which continues as long as there are events occurring in the circuit. After the elapse of a sufficiently long time following the last primary input event, when there are no events in the circuit, the circuit is considered to be in a *steady* state.

It is a property of combinational logic that the steady state signals are uniquely determined by the input signals. The signal values in the transition interval do not have that property, since the instantaneous value of a signal may also depend on delays in the circuit. We will call the signal values in steady state as the *correct logic values*. During the transition interval, a signal can have multiple events. However, it will always settle to the correct logic value.

Consider a Boolean gate in a combinational CMOS circuit. The gate has a single output and can have one or more inputs. We assume that the Boolean gate consumes energy only when it produces an event at its output.

Theorem 1: For a correct operation with minimum energy consumption, a Boolean gate must produce no more than one event at its output during a transition interval.

Proof: Consider the gate output signal before and after the transient interval. Let us denote these two steady state values as a and b, each of which can be either 0

0-7695-0013-7/99 $10.00

or 1. We consider two cases:

- Case 1: $a = b$. If the gate produces events at its output in response to one or more events that occur at its inputs, then it must produce an even number of events (0, 2, 4, . . .) This is because, the output, which begins with value a, must return to the same value. For this case, minimum energy is consumed when the output has no event.
- Case 2: $a \neq b$. When the gate responds to input events, the number of events at its output must be odd (1, 3, 5, . . .) This is because, the steady state output value must differ from a. For this case, minimum energy is consumed when the output has one event.

Combining the two cases, we find that for arbitrary signal changes, minimum energy is consumed when the gate has either no event or just one event at the output. The correct steady state logic value, if obtained with two or more events, will consume more energy. ■

Boolean gates have no memory. Their output is uniquely determined by inputs. The correct output is determined by the correct (steady state) inputs. However, in the transition interval, a gate may be quick to respond to extra events. According to Theorem 1, the number of *essential* events is either 0 or 1. The *extra* events always occur in pairs (see proof of Theorem 1); the two events in the pair cancel each other. Next, we examine the maximum number of events that can occur at the output of a gate. We will assume the gate to have an ideal delay d as defined below:

Definition (Zero-Delay): An event at a gate input, which changes an input value in such a way that the output logic value should differ from the prevailing output, causes the zero-delay gate to produce an output event. The output event occurs at the same time as the time of the input event. We will call such an output event a *zero-delay event*, since it coincides in time with its "cause" event.

Definition (Ideal Delay): If one or more events occur at time t at the input of a gate with ideal delay d such that a zero-delay gate would have produced an event e, then the gate will produce e at time $t + d$, provided no other event occurs at the input in the interval $[t, t + d]$, requiring another zero-delay event. This is shown as the single event case in Figure 1(a). In general, any number of zero-delay events can occur in an interval of duration d. As shown in Figure 1(b), an even number of zero-delay events in an interval $[t, t+d]$ will produce no event at the output of a gate with ideal delay d. This is because each alternate event is canceled by the following event. For an odd number of events in the interval $[t, t + d]$, however,

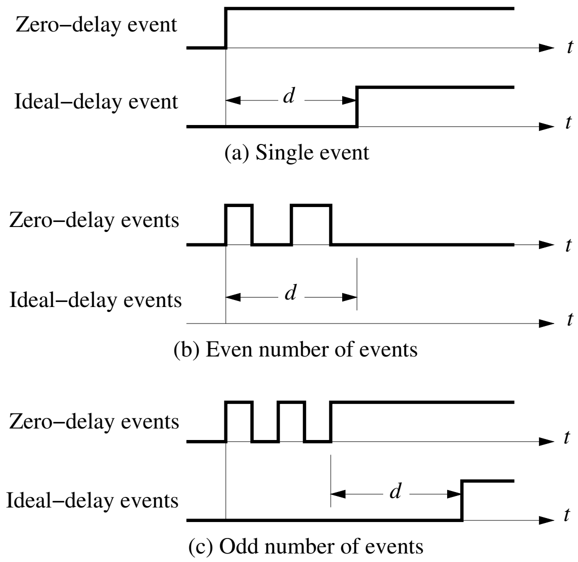

Figure 1: Events produced by an ideal-delay gate

the last event is not canceled. After the last event, if no zero-delay event occurs for an interval d, then an output event is produced by the ideal-delay gate, as shown in Figure 1(c).

Theorem 2: Given that events at the inputs of a gate occur at times, $t_1 \leq t_2 \leq \ldots \leq t_n$, the number of events at the output of the gate cannot exceed

$$\min(n,\ 1 + \lfloor \frac{t_n - t_1}{d} \rfloor) \tag{1}$$

where d is the ideal delay of the gate.

Proof: We will derive two upper bounds on the number of ideal-delay output events produced by the gate. The lower of these two upper bounds will then be an acceptable tighter upper bound.

First Upper Bound. Any event at the input of a gate can potentially produce a zero-delay event at the output. Since an output event must be caused by some input event, the number of output events cannot exceed n, which is the first upper bound on the number of events.

Second Upper Bound. We will consider the case where n zero-delay events are arbitrarily placed in an interval of duration $t_n - t_1$. In Figure 2, these potential zero-delay events are shown by upward pointing arrows. From the definition of ideal-delay, we find that a zero-delay event at t_i can produce an ideal-delay event only if $t_{i+1} - t_i \geq d$. In other words, the *gap* between successive events must not be less than d. Since the events can occur at arbitrary times, the gap d can only be guaranteed for the last event at t_n. The number of output events cannot exceed the maximum number of gaps of duration d that

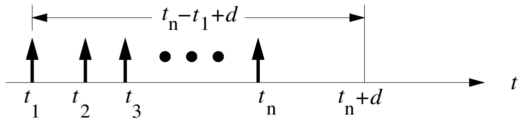

Figure 2: An arbitrary distribution of n events

can be placed in the interval $[t_1, t_n + d]$. This is given by

$$\lfloor \frac{t_n + d - t_1}{d} \rfloor \; = \; 1 + \lfloor \frac{t_n - t_1}{d} \rfloor \qquad (2)$$

This is the second upper bound on the number of output events.

Combining the two upper bounds, we get the tighter upper bound of Equation (1). ■

Theorem 2 provides an insight into reducing the number of output events. The upper bound of expression (1) takes the smallest value when

$$t_n - t_1 \;\le\; d \qquad (3)$$

This upper bound is 1, which according to Theorem 1 is the condition for the correct operation with minimum energy consumption.

Definition: A combinational circuit has a **minimum transient energy design** if for any set of arbitrary simultaneous changes at primary inputs, steady state is reached with no more than one signal transition per gate output.

Inequality (3) is a generalized condition for minimum transient energy design. Two possible ways of design are discussed below.

Balanced Delay. If all input events occur simultaneously, i.e., $t_1 = t_2 = \;\ldots\; = t_n$, then inequality (3) is satisfied for any value of d. This can be accomplished by equalizing the delays of signal paths arriving at the gate. In practice, gates on fast paths can be slowed down. Also, a fact that is often not recognized is that path delays do not have to be perfectly balanced. According to inequality (3), *as long as the differences between path delays are within the gate delay (d), hazards will not be produced.* However, difficulties arise in certain cases:

- In general, the delays of rising and falling events can be different. Balancing of both types of delays for a path then becomes difficult.
- In some cases, slowing of a path requires extra delay buffers. Examples are primary input signal and fanout signals that feed into gates at different levels.

Hazard Filter. This is done by increasing the delay of the gate such that $d \;\ge\; t_n - t_1$. Essentially, a large delay gate acts like a low-pass filter that eliminates the rapid variations within the transition interval. In practice, the delay of gates can be increased without adding buffers. However, a disadvantage of the method is that it increases the overall delay of the circuit [1].

Definition: The delay of a path is the sum of delays of all elements (gates and interconnects) on the path. For a combinational circuit the **overall circuit delay** is the longest delay on any path between primary inputs and primary outputs.

3. Minimum Transient Energy Design

For a given logic circuit, we can achieve a minimum transient energy design if inequality (3) is satisfied for all gates. Such a design requires adjustment of delays and may sometimes increase the overall delay of the circuit. Delays of logic circuit elements (gates and interconnecting nets) can be lumped as output and input delays for gates. The *output delay* of a gate consists of the switching delay of the gate, and the time of charge or discharge of the lumped output capacitance. The output of the gate may feed into several gates through a fanout net. The output delay is associated with the stem of the fanout. In general, fanout lines will have different lengths and the signal will arrive at each fanout at a different time. This effect is represented by the *input delay* of the fanout gates. Thus, a gate will have one delay associated with its output and separate delays for each input.

The output delay can be controlled by varying the sizes of devices in the gate. A control of input delays is more difficult as it involves changing the geometry of nets. For a given design, therefore, input delays are changed by inserting buffers. In an energy minimized design, the use of such buffers in not desirable since each buffer will consume additional energy. Before we formulate the problem of delay adjustment, the following result is useful.

Theorem 3: If the overall circuit delay is constrained not to increase, then a minimum transient energy design cannot be guaranteed by only increasing the output delays.

This result can be proved by contradiction. The example of Figure 3 shows a circuit with two longest paths of equal delays. Each block has one unit of minimum delay and interconnects are assumed to have no delay. The original design does not contain block C. So the output of A feeds into B. Inequality (3) can be satisfied for gate B in two ways. Either the output delay of A is increased to 3 units, or the delay of B is increased to 3 units. In both cases the overall circuit delay is increased. If the overall delay is to be held fixed at 5 units, then we must increase the input delays for B by a 2 unit delay buffer C

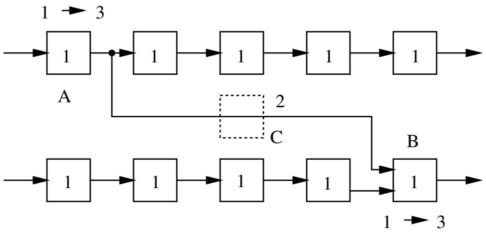

Figure 3: An example for Theorem 3

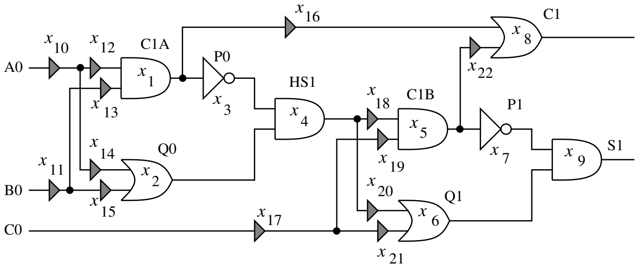

Figure 4: Linear program variables for *add1b* circuit

shown with broken lines. This is an example where minimum transient energy design was possible only with an increase in the overall delay. There are other examples where primary input signals are involved and one must increase input delays to satisfy inequality (3).

Notice that an increase of an input delay requires the insertion of a buffer. Thus, any attempt at minimizing energy by the manipulation of input delays has an associated penalty of the energy consumed in the buffer. According to Theorem 3, if overall circuit delay was constrained, the use of buffers may be necessary. In other words, when circuit speed is most critical, the circuit may have to use more energy than the possible absolute minimum energy operation.

Suppose, we restrict our attention to the minimum transient energy design where no constraint is imposed on the overall delay. Our problem is then to obtain the minimum transient energy design for the smallest increase in the overall delay without inserting any delay buffers. For a general case, the following result holds.

Theorem 4: If the overall delay of the circuit is allowed to increase, then a minimum transient energy design is always possible by adjusting the output delays of gates. Such a design will have the smallest transient energy consumption among all delay transformations of the original circuit.

The proof is straightforward, since inequality (3) can be satisfied for all gates by only increasing the right hand side. A simple procedure would be to levelize the circuit from primary inputs to primary outputs. Then we process gates in increasing level order and, when necessary, increase output delays to satisfy inequality (3). For example, for the circuit of Figure 3, only the delay of B will be increased to 3 units. Since no buffer is added, we obtain the minimum transient energy design.

4. Linear Programming

A linear program determines a set of variables such that an objective function is minimized (or maximized) under given constraints [5]. We give an illustrative linear programming model, which can be suitably modified for specific technologies and layout styles.

Variables: We have two types of variables: gate delays and buffer delays. Gate delay variables, one per gate, are the inertial delays of gate outputs. Buffer delays are the delays of buffers placed on primary inputs and all fanout lines. Each gate delay is assumed to have a minimum value of *unity*. The minimum delay for a buffer is zero. Consider the adder circuit *add1b* shown in Figure 4. Gate delays are x_1 through x_9 and the delays of buffers (shaded elements) are x_{10} through x_{22}. Note that no buffer is placed on a fanout feeding into a single-input gate. This is because the buffer delay can be included in the gate delay. For example, consider the fanouts of gate C1A. One fanout has a buffer of delay x_{16} and the delay of the other fanout is controlled by by the inverter P0. The lower bounds on variables are:

$$x_i \geq 1 \text{ for } 1 \leq i \leq 9 \quad \text{and} \quad x_i \geq 0 \text{ for } 10 \leq i \leq 22 \tag{4}$$

In general, any minimum delays can be specified for gates.

Objective function: Since we want a solution with the smallest number of buffers, a suitable objective function to be minimized is the sum of all buffer delays. For the circuit of Figure 4, we minimize

$$\sum_{i=10}^{22} x_i \tag{5}$$

Note that our *real* objective is to minimize the number of buffers. However, expression (5) is preferred since it is a linear function of variables. In general, any weighted combination of (5) and path delays can be minimized.

Constraints: There are two sets of constraints. The first set specifies the requirement of the overall delay of the circuit. Thus the delay of all, or some selected subset of, paths should be bounded by a user-specified parameter *maxdel*. For the *add1b* circuit (Figure 4), constraints on two of the paths are shown below:

$$x_{10}+x_{12}+x_1+x_3+x_4+x_{18}+x_5+x_{22}+x_8 \leq maxdel \tag{6}$$

$$x_{10}+x_{12}+x_1+x_3+x_4+x_{18}+x_5+x_7+x_9 \leq maxdel \tag{7}$$

Table 1: Minimum transient energy design of the *add1b* circuit using AMPL [5]

Given delay *maxdel*	Linear Programming solution (all other variables are 0)											No. of buffers	Max. IO delay at	
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{16}	x_{17}		C1	S1
6.0	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**2.0**	**1.0**	**2.0**	**1.0**	**1.0**	**2.0**	**2**	**6.0**	**6.0**
6.5	1.0	1.0	1.0	1.0	1.0	2.0	1.0	2.5	1.0	0.5	2.0	2	6.5	6.0
7.0	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**2.0**	**1.0**	**3.0**	**1.0**	**0.0**	**2.0**	**1**	**7.0**	**6.0**
8.0	1.0	1.0	1.0	1.0	1.5	2.5	1.0	3.5	1.5	0.0	1.5	1	8.0	7.0
9.0	1.0	1.0	1.0	1.0	2.0	3.0	1.0	4.0	2.0	0.0	1.0	1	9.0	8.0
10.0	1.0	1.0	1.0	1.0	2.5	3.5	1.0	4.5	2.5	0.0	0.5	1	10.0	9.0
$\geq$ **11.0**	**1.0**	**1.0**	**1.0**	**1.0**	**3.0**	**4.0**	**1.0**	**5.0**	**3.0**	**0.0**	**0.0**	**0**	**11.0**	**10.0**

Similar constraints can be applied to all IO paths. However, note that other paths differ from these two paths in segments whose delays are controlled by the second set of constraints.

The second set of constraints specify the inequality (3) for each pair of paths converging at a gate. These constraints must be specified for all multi-input gates. Suppose the delay of gate i is specified as x_i and the delays of two paths incident on two *different*[1] inputs of this gate are d_1 and d_2. These delays are represented as sums of delays of gates and buffers on paths. Then, the linear constraints of inequality (3) are:

$$d_1 - d_2 \leq x_i \quad \text{and} \quad d_2 - d_1 \leq x_i \qquad (8)$$

Typical constraints for gate HS1 in Figure 4 for a pair of paths are:

$$x_{10} + x_{12} + x_1 + x_3 - x_{11} - x_{15} - x_2 \leq x_4 \qquad (9)$$

$$-x_{10} - x_{12} - x_1 - x_3 + x_{11} + x_{15} + x_2 \leq x_4 \qquad (10)$$

Similar constraints must be specified for all path-pairs converging on two different inputs of HS1, as well as for all multi-input gates C1A, Q0, C1B, Q1, C1 and S1.

5. Results

We wrote a C program to derive linear programming models in AMPL [5] directly from the netlist of the circuit. AMPL is a mathematical programming language. Its compiler produces a linear program whose solution is found by the MINOS 5.5 solver. In general, the minimum delays of gates can be specified according to design considerations. For our examples we assume that all gates have a minimum delay of unity and all buffers have minimum delay of zero.

For the *add1b* circuit, the AMPL model contains 17 path delay constraints of the type (6) and (7). There are 49 pairs of constraints of the type (8) for all two-input gates. Since the minimum IO delay is 6 units, no solution is possible for $maxdel < 6$. For other values of $maxdel$, the solutions are shown in Table 1. When $maxdel = 6$ is specified, we require two buffers. For $maxdel = 7$ only one buffer is needed and for $maxdel \geq 11$ no buffer is needed. In all cases, the condition of inequality (3) was satisfied at all multi-input gates by optimally combining delay balancing and hazard filtering to meet the maximum overall delay ($maxdel$) requirement. Representative signal waveforms, as obtained by Spice simulation, were similar to those reported in our earlier paper [1]. These waveforms clearly show that no gate has more than one transition per vector, a necessary condition for the optimum design (Theorem 1).

Notice that the linear objective function of Expression 5 minimizes the sum of buffer delays instead of the number of buffers. The linear program, therefore, does not always find the minimum buffer solution. However, as the value of $maxdel$ is increased, the solutions at which the number of buffers just drops to a lower value are optimum. These are shown in bold in Table 1.

Power Estimation. We estimated the peak and average power consumed by the *add1b* circuit. These estimates are obtained by the techniques described by Hsiao *et al.* [6, 7]. In their method, a genetic algorithm finds a vector-pair that produces the largest number of signal transitions in the circuit with given gate delays. The number of transitions is then determined by an event-driven logic simulator. This number is normalized as power by multiplying with the number of capacitive nodes. The average power is determined by normalizing the per-vector activity averaged over a large number of random vectors (10,240 for our estimates) simulated in the event-driven mode. Table 2 gives the peak and average power estimates for the three optimized cases of the *add1b* circuit (shown in boldface in Table 1.) The estimates are comparative to two reference cases, which correspond to the original buffer-less circuit. The reference

[1] Note that two paths incident on the same input of a gate cannot produce a hazard.

Table 2: Power estimation for optimized *add1b* circuits

$maxdel$	No. of buf.	Power with respect to Ref.			
		Ref: model delays		Ref: unit delays	
		Peak	Ave.	Peak	Ave.
6	2	0.60	0.89	0.60	0.90
7	1	0.56	0.85	0.56	0.86
≥ 11	0	0.52	0.80	0.52	0.81

Table 3: Minimum transient energy *alu4* circuits

$maxdel$	No. of buf.	Power with respect to Ref.			
		Ref: model delays		Ref: unit delays	
		Peak	Ave.	Peak	Ave.
7	5	0.55	0.76	0.74	0.82
10	2	0.54	0.75	0.73	0.80
12	1	0.54	0.73	0.72	0.79
≥ 15	0	0.53	0.73	0.71	0.79

with "model delays" has gate delays scaled in proportion to fanouts. In the other reference, all gates have unit delays. All optimized circuits have a maximum of one transition per gate per vector and their power estimates are the same as those in the zero-delay case. The differences between the optimized circuits are due to different numbers of buffers, which basically increase the number of capacitive nodes in the circuit. Although this circuit is small and variations in power are not large, these results clearly show the trade-off between the reduced transient power and the increase of the overall circuit delay. In actual applications, one expects the consumption to be between the peak and average estimates.

A Four-Bit ALU Circuit. Our second example is *alu4* [8]. This circuit has 14 inputs, 8 outputs and 58 gates. The gates were assumed to have a minimum delay of one unit and 129 buffers of zero minimum delay were inserted. The AMPL model contains 457 path delay constraints and over 20,000 multi-input gate constraints. A linear programming solution takes less than 200 seconds on an SGI Octane (MIPS R10000) computer. The longest path has seven gates. For $maxdel = 7$, we need 5 buffers and no buffer is needed for $maxdel \geq 15$. See Table 3. Compared to the model delay reference, we notice that the minimum transient energy design (with no buffer) consumes 53% peak and 73% average power.

6. Conclusion

The main result of this work is a generalized condition for the minimum transient energy. A linear programming technique is shown to achieve this condition for all gates optimally under the overall delay constraint for the circuit. Thus, the problem of transistor sizing [4] can be reformulated for simultaneous optimization of area, delay and power. The conditions for the linear program require path enumeration. Since the constraints on various gates are not independent, further analysis may reduce the total number of constraints. Another possibility is to partition the circuit and optimize each partition for minimum transient energy.

Combinational circuit analysis is valid for synchronous systems. For asynchronous circuits, cycle delays are important and the analysis will require modifications.

Increased switching delay can influence the short circuit current, which flows through the momentary short between the power supply and ground created during switching. The energy consumption due to this current is usually neglected. However, attention should be focused on this component for large delay gates.

The techniques in this paper could have applications in minimizing the CPU time of software and in reducing the cost in project management.

Acknowledgment – The authors thank D. M. Gay and B. W. Kernighan of Bell Labs for help with AMPL and M. Hsiao of Rutgers University for providing the power estimation programs.

References

[1] V. D. Agrawal, "Low-Power Design by Hazard Filtering," in *Proc. 10th International Conf. VLSI Design*, 1997, pp. 193–197.

[2] A. Bellaouar and M. I. Elmasry, *Low-Power Digital VLSI Design: Circuits and Systems.* Boston: Kluwer Academic Publishers, 1995.

[3] A. P. Chandrakasan and R. W. Brodersen, *Low Power Digital CMOS Design.* Boston: Kluwer Academic Publishers, 1995.

[4] J. P. Fishburn and A. E. Dunlop, "TILOS: A Posynomial Programming Approach to Transistor Sizing," in *Proc. International Conf. CAD*, 1985, pp. 326–328.

[5] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming.* South San Francisco, California: The Scientific Press, 1993.

[6] M. S. Hsiao, E. M. Rudnick, and J. H. Patel, "Effects of Delay Models on Peak Power Estimation of VLSI Sequential Circuits," in *Proc. IEEE/ACM International Conf. Computer-Aided Design*, November 1997, pp. 45–51.

[7] M. S. Hsiao, E. M. Rudnick, and J. H. Patel, "K2: An Estimator for Peak Sustainable Power of VLSI Circuits," in *Proc. International Symp. Low Power Electronics and Design*, 1997, pp. 178–183.

[8] Texas Instruments, Inc., Dallas, Texas, *TTL Data Book for Design Engineers*, 2 edition, 1976. p. 7-280.