Desensitization for Power Reduction in Sequential Circuits

Xiangfeng Chen, Peicheng Pan, C. L. Liu
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801

Abstract

In this paper, we describe a technique for power reduction in sequential circuits. Existing signals in the circuit are used to selectively disable some of the registers so that a portion of the circuit will be deactivated. Consequently, average power consumption in the circuit is reduced at a cost of small increases in area and delay. We present an algorithm for determining the desensitizing signal for each register. A significant amount of power reduction is achieved in a number of benchmark circuits according to our experimental results.

1 Introduction

Reduction in average power consumption is an increasingly important objective in integrated circuit (IC) design nowadays. One reason is the requirement of longer battery life for the growing class of personal computing devices, such as portable desktops and wireless communication equipments. Another reason is the dramatic decrease in chip size and increase in transistor count and clock rate for integrated circuits. High power consumption increases the cost to handle heat dissipation and diminishes the reliability of the circuits.

Power reduction can be performed at various design levels, including circuit, logic, register-transfer, behavior, and system levels [5]. In CMOS technology, over 90% of the power consumed in a digital circuit is due to the switching activities in the circuit [4], since there is no significant current flow in a CMOS circuit when it is in the quiescent state. Consequently, *transition density* is an important factor in determining the average power dissipation in a CMOS circuit [9, 6, 8, 13]. For sequential circuits, one of the design techniques is to deactivate a portion of a circuit by selectively disabling some of the registers in the circuit so that power consumption will be reduced. For instance, a *precomputation* approach was proposed in [2], where the primary output values are precomputed for a subset of the input combinations one clock cycle in advance. A portion of the combinational circuit can be deactivated by disabling some of the inputs. In [3], a technique was proposed to detect *self-loops* in a finite state Mealy machine. When the machine is in a self-loop, all the registers can be disabled so that switching activity in the circuit will be reduced.

*The work was partially supported by the National Science Foundation under grant MIP-9222408.

In this paper, we propose another technique which makes use of some of the existing signals in a circuit to selectively disable some of the registers. Consequently, a portion of the circuit can be deactivated with the logic overhead being rather small. Our approach is based on the concept of the *controlling values* of the combinational elements in the circuit. For example,

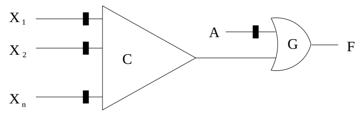

Figure 1: An example

consider the circuit in Figure 1. Assume that C is a combinational circuit with only one output. Also, assume that each register has a load disable pin. A register is disabled when the load disable signal is 1, and is enabled when the load disable signal is 0. Suppose that connections are added from A to the load disable pins (LDs) of the registers at the inputs of C as shown in Figure 2. When $A = 0$, the registers are enabled and, clearly, the functionality of the circuit does not change. When $A = 1$, which is the *controlling value* of the OR gate G, the output of the circuit F will be equal to 1 regardless of the output value of C. Since the registers at the inputs of C will be disabled, switching activities in C will be eliminated in the subsequent cycle. In general, if subcircuit C has more

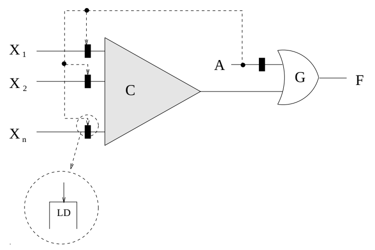

Figure 2: Illustration of desensitization

than one output and the values of some of the outputs are used in other parts of the circuit, then not all the registers at the inputs of C can be disabled. However, a register can still be disabled by the signal at A if the output signal of the register must pass through the OR gate G in order to reach any primary output of the circuit. Consequently, a portion of C (shaded area in Figure 3) can be deactivated.

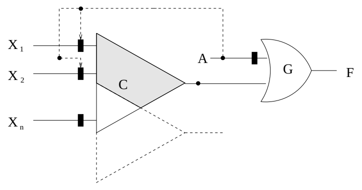

Figure 3: Subcircuit with more than one outputs

In a sequential circuit, a register r is said to be *desensitizable* by a signal A if all of primary outputs of the circuit remain unchanged when A is connected to the load disable pin of r. A is called a *desensitizing* signal for r. Given a sequential circuit at the logic level, we present an algorithm for determining the desensitizing signals for all the registers. The algorithm is based on a breadth-first traversal of the circuit and has complexity $O(|E|)$, where $|E|$ is the number of interconnections in the circuit. Up to 34% power reduction is attained for many MCNC benchmarks with only marginal increases in area and delay.

The remainder of this paper is organized as follows. Section 2 introduces the structural model of a sequential circuit and the fundamental concepts used in this paper. Section 3 presents the desensitization method. Section 4 gives an algorithm for determining the desensitizing signal for each register. Section 5 presents some experimental results. Section 6 discusses a number of generalizations of the technique. Section 7 concludes with future work.

2 Preliminaries

In this section, we introduce the notations and terminologies used throughout this paper. A synchronous sequential circuit can be represented as a directed multigraph. A *vertex* represents a primary input, a primary output, a combinational element, or a register. An arc represents an interconnection between two vertices.

An input A to a combinational element is called a *controlling (desensitizing) input* with a *controlling (desensitizing) value* c if the output of the combinational element is independent of the values of the other inputs when $A = c$. We say that c is the controlling value of a combinational element G if all the inputs of G are controlling inputs with c being the controlling value. For example, 1 is the controlling value of an OR or a NOR gate, and 0 is the controlling value of an AND or a NAND gate.

All registers in a sequential circuit have the following characteristics: Each register has a single input and a single output. All registers are clocked by the same periodic wave form. At each clock tick, a register samples its input and the sampled value will be available at the output at the next clock tick. A register has a *load disable* pin which can be implemented by the *gated-clock* technique [10]. If the load disable signal is equal to 0, the register is enabled and the current input value will be the output value of the register at the next clock cycle. If the load disable signal is equal to 1, the register is disabled and the output value of the register will remain the same as that in the preceding clock cycle. An example of such registers is an edge-triggered D-type flip-flop with a disable pin [14]. We also assume that there is at least one register in any cycle in the circuit. The reason for such an assumption is to avoid the problems of asynchronous latching, oscillation, and racing conditions.

A *path* is a sequence of vertices and arcs. A vertex v_1 is said to be a *dominator* of a vertex v if all the paths from v to the primary outputs of the network must pass through v_1.

For example, in Figure 4, $G0$ is a dominator of $X2$, $Y2$ and register $R1$, $R2$, and $R3$.

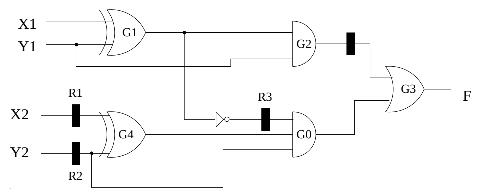

Figure 4: An example

A set of vertices $S = \{v_1, v_2, ..., v_k\}$ is said to be a *dominator set* of a vertex v if any path from v to a primary output of the network must pass through one of the vertices $v_1, v_2, ..., v_k$. S is called a *minimal dominator set* of v if any subset of S is not a dominator set of v.

For example, in Figure 4, $\{G0, G2, G3\}$ is a dominate set of $X1$. $\{G0, G2\}$ is a minimal dominate set of $X1$, so is $\{G3\}$.

3 Desensitization

As was illustrated in Section 1, desensitization is a technique that can be employed to selectively disable some of the registers in a sequential circuit without changing the functionality of the circuit. In this section, we describe how the desensitizing signals for the registers can be determined.

3.1 Determining Desensitizing Signal

Theorem 3.1 Given a combinational element G in a sequential circuit N, if v is a controlling input of G with a controlling value c, and there is a register between v and G, then any register r is desensitizable by the signal $v \oplus c$ if G dominates r and there is no register on any path from r to G.

Proof: We claim that the output of G is unchanged if r is desensitized by the signal $v \oplus c$.

(1) If the value of v is not equal to the controlling value of G, i.e. $v \neq c$, then $v \oplus c = 1$. Thus, r will not be disabled and the behavior of the network is unchanged.

(2) Suppose that at cycle t_0, the value of v is equal to the controlling value of G, i.e. $v = c$, then $v \oplus c = 0$ and r is disabled. v will arrive at G at cycle $t_0 + 1$. Since there is no register between r and G, the input of r at cycle t_0 *should* also arrive at G at cycle $t_0 + 1$. However, this input is a *don't care* input since v is equal to the controlling value of G and G dominates r. Thus, r can be safely disabled. $\square$

For the example in Figure 4, register $R1$, $R2$ can be desensitized by the output signal of $G1$, and $R3$ can be desensitized by the output signal of $G2$, as shown in Fig 5. In this example, the average power consumption is reduced by 20.3% after these three registers are desensitized. Power consumption was computed using SIS-1.2 [12].

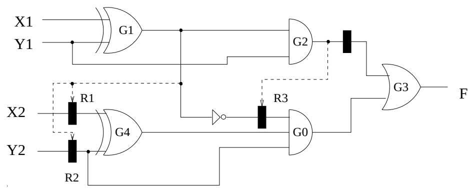

Figure 5: An example of desensitization

Theorem 3.1 can be applied when the register to be desensitized, r, is dominated by a combinational element G. If r is not dominated by a single combinational element, a minimal dominator set S of r can be found, and the register is desensitizable by a conjunction of signals obtained from the controlling inputs of the vertices in S.

Theorem 3.2 Let r be a register in a sequential circuit N. Let $S = \{v_1, v_2, ..., v_k\}$ be a minimal dominator set of r. Suppose that for each v_i in S u_i is a controlling input with a controlling value c_i, and there is a register between u_i and v_i. If there is no register on any path from r to v_i for each v_i, then r is desensitizable by the signal $\wedge_{i=1}^{k} u_i \oplus c_i$.

Proof: Similar to the proof of Theorem 3.1.

If a register r has two or more minimal dominator sets, then r is desensitizable by the disjunction of the desensitizing signals corresponding to the minimal dominator sets. Note that a disjunction of the desensitizing signal will lead to further reduction in switching activity.

Theorem 3.3 Let r be a register in a sequential circuit N. Let $S_1, S_2, ..., S_k$ be minimal dominator sets of r. Let f_i be the desensitizing signal corresponding to S_i (computed according to Theorem 3.2), then r is desensitizable by the signal $\vee_{i=1}^{k} f_i$.

Proof: Immediate from Theorem 3.2.

For example, for the circuit shown in Figure 4, register $R1$ and $R2$ have two minimal dominator sets $\{G0\}$ and $\{G3\}$. Since the desensitizing signal for $R1$ and $R2$ corresponding to the minimal dominator set $\{G0\}$ is the output signal of $G1$ and the desensitizing signal of the minimal dominator set $\{G3\}$ is the output signal of $G2$, $R1$ and $R2$ can be desensitized by the disjunction of the output signals of $G1$ and $G2$, as shown in Figure 6. In this case, the average power consumption is reduced by 31.1% (in comparison with the 20.3% reduction for the circuit in Figure 5).

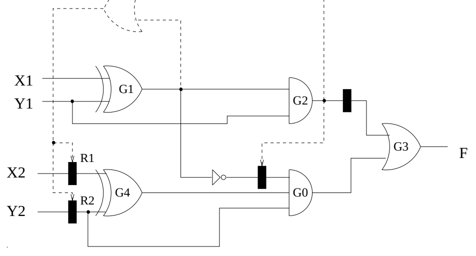

Figure 6: Desensitizing signal from two dominator sets

3.2 Cross Control

Desensitization may introduce hazards in the circuit. For example, for the circuit in Figure 7, the output of gate G is undetermined and will depend on the race condition. If at clock cycle t_0, $A = B = 1$, then both $F1$ and $F2$ are enabled and $G = 1$. If at the next clock cycle $t_0 + 1$, $A = B = 0$ and the desensitizing signals arrive at the load disable pins of the registers before the register inputs do, then both $F1$ and $F2$ are disabled and $G = 1$ which is not the correct value. In this case, only one of the desensitizing connections can be made.

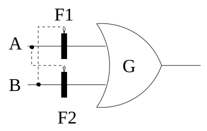

Figure 7: A trivial cross control

Definition 3.1 (*Cross Control*) Let r_1 and r_2 be two registers in a sequential circuit N. If r_1 is desensitized by the input signal of r_2, and r_2 is desensitized by the input signal of r_1, then these two connections form a pair of *cross controls* in the circuit. If the outputs of r_1 and r_2 are fanins of the same combinational element, then they form a pair of *trivial cross controls*, otherwise, they form a pair of *non-trivial cross controls*.

Non-trivial cross controls will also yield hazards. For example, the outputs of $G1$ and $G2$ in the circuit shown in Figure 8 are undetermined.

Theorem 3.4 No two desensitizing signals determined according to Theorem 3.1 form a pair of non-trivial cross controls.

Proof: If there is a pair of non-trivial cross controls between registers A and B as shown in Figure 8, then according to Theorem 3.1, there is a path from $G1$ to $G2$ and vise versa. Since neither of these paths contains any register, we have a cyclic path in the circuit that contains no register, which contradicts the assumption in section 2. □

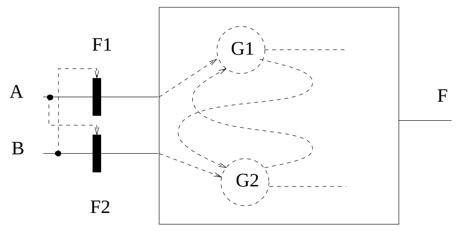

Figure 8: A non-trivial cross control

Since the algorithm described in section 4 is based on Theorem 3.1, no non-trivial cross control pairs will be generated by the algorithm. Trivial cross control pairs are prevented when only one of the fanins to a combinational element will be selected as a desensitizing signal.

4 An Algorithm for Determining Desensitizing Signals

In this section, we describe an algorithm for determining a desensitizing signal for every register in a sequential circuit. In the directed multigraph that represents a sequential circuit, each arc a is associated with a function $f(a)$ which is referred to *desensitizing function* for a. Given a register r, let $a_1, a_2, ..., a_k$ be the fanout arcs of r and $f(a_i)$ be the corresponding desensitizing functions for a_i. Then r is desensitizable by $\wedge_{i=1}^{k} f(a_i)$. By traversing the graph from the primary outputs to the primary inputs, desensitizing functions for all arcs are determined. The time complexity for traversing all the arcs is $O(|E|)$, where $|E|$ is the number of arcs in the graph.

4.1 Desensitizing Functions for the Arcs

A sequential circuit is represented by a multigraph $N = (V, A)$, where V and A are the sets of vertices and arcs, respectively. Initially, a constant 0 is assigned to each arc as its desensitizing function. These functions are updated when the vertices are visited in a breadth-first search from the primary outputs to the primary inputs. A *first-in first-out* queue Q is used to implement the breadth-first search. For each arc $a \in A$, $f(a)$ is the desensitizing function for a when the procedure terminates.

```
DESENSITIZE($N$)
1. for each arc $a \in A$ do
2.     $f(a) \leftarrow 0$
3. for vertex $v \in V$ do
4.     visited[$v$] $\leftarrow$ FALSE
5.     if $v$ is a primary output then
6.         add_queue($v$, $Q$)
7. while $Q \neq \phi$ do
8.         $v \leftarrow$ get_vertex($Q$)
9.         if visited[$v$] = FALSE then
10.            if $v$ is a register then
11.                visited[$v$] $\leftarrow$ TRUE
12.                $a \leftarrow$ fanin arc of $v$
13.                $f(a) \leftarrow 0$
14.                $u \leftarrow$ fanin vertex of $v$
15.                add_queue($u$, $Q$)
16.            else for each fanout arc a of $v$ do
17.                    $f \leftarrow f \wedge f(a)$
18.                    if $v$ has at least one fanin from a
                           register then
19.                        $u \leftarrow$ choose_one_fanin($v$)
20.                        for each fanin arc $a$ of $v$ do
21.                            if $a$ is from $u$ then
22.                                $f(a) \leftarrow f$
23.                            else $f(a) \leftarrow u \vee f$
24.                    else for each fanin arc $a$ of $v$ do
25.                            $f(a) \leftarrow f$
26.                    for each fanin vertex $u$ of $v$ do
27.                            add_queue($u$, $Q$)
28.            visited[$v$] $\leftarrow$ TRUE
```

The procedure works as follows. The desensitizing function associated with each arc is initialized to be 0 in line 1-2. Lines 3-6 mark all the vertices as unvisited, and place all the primary output vertices in Q. Line 7-28 constitute the main loop of the procedure. The loop iterates as long as there are vertices that have not been visited. Line 8 gets a vertex from queue Q. If this vertex has been visited, then lines 9-28 are skipped. Otherwise, if the vertex represents a register, 0 will be assigned as the desensitizing function for its fanin arc. Lines 16-17 compute f as the conjunction of all the desensitizing functions of the fanout arcs of v if v represents a combinational element. If some of v's fanins are outputs of registers, line 19 selects one of the registers and denotes the input of it by u. Line 22 assigns f to be the desensitizing function of the fanin arc from u. Line 23 assigns $u \vee f$ to be the desensitizing functions of the fanin arcs not from u. Lines 24-25 assign f to be the desensitizing functions of all the fanins if none of them is through a register. Lines 26-27 put all the fanin vertices to the queue Q. Line 28 marks v to be visited.

We have the following theorem.

Theorem 4.1 Let r be a register in the circuit. Suppose that r has k fanouts. Let $f_1, f_2, ..., f_k$ be the desensitizing functions for the fanout arcs, then r can be desensitized by the signal $\wedge_{i=1}^{k} f_i$.

4.2 Register Replication

As was pointed out above, a register with k fanouts can be desensitized by the signal $\wedge_{i=1}^{k} f_i$, where $f_1, f_2, ..., f_k$ are the desensitizing functions for the fanout arcs of the registers. However, the probablity $\mathcal{P}(\wedge_{i=1}^{k} f_i = 1)$ can be very small and the power consumption in the extra control logic can be higher than the power reduction in the original circuit. In this case, the register can be replicated so that different copies of it have different desensitizing signal.

The area of the circuit is increased after register replication. It may also increase the total capacitance of the registers, one of the factors determining power consumption. Although such increase will be off-set by the fact that each of the replicated register has fewer fanouts than the original register, extensive register replication may increase the power consumption. It is a part of the ongoing work to replicate the registers optimaly.

5 Experimental Results

We have implemented the algorithm in Section 4 and tested it on a number of MCNC benchmark circuits. For the sake of simplicity, the circuits were mapped using a cell library of simple logic gates so that the controlling values of the combinational elements can be determined easily. The experimental results are presented in Table 1 and Table 2, in which the number of inputs (I), outputs (O), and registers (R) of the circuits are listed. The number of literals, levels, and average power consumption are shown for both the original circuit and the circuit with desensitization. The power estimation package in SIS-1.2 was used, and a zero delay model and a clock rate of 20 MHz are assumed.

We first take a number of combinational benchmark circuits and add registers in the circuits. In some cases, registers are added only at the inputs and outputs, while in other cases, registers are added randomly. The results are presented in Table 1. Average power consumption decreases in all circuits with desensitization. The area and delay penalty are indicated by the increase in the number of literals and levels in the circuit. They are very small in almost all of the circuits. For example, in the case of `mux`, we attained a 34% power reduction without increasing the number of literals or levels.

Table 2 presents the results for sequential benchmark circuits. Some of the registers were replicated to maximize the reduction in switching activity. Again, power reduction is attained in all the circuits with small increase in area and delay. Since power estimation requires substantial computation resources (time and storage), most of the circuit tested are finite state machines in the MCNC benchmark suite. However, we

Circuit				Original			With DS			% Power
Name	I	O	R	Lits	Lvls	Power(μw)	Lits	Lvls	Power	Reduction
5xp1	7	10	35	392	10	2148.6	439	10	1704.0	20.7
9symml	9	1	19	462	13	2005.0	487	14	1667.5	16.8
clip	9	5	77	732	12	4382.2	823	12	3412.6	22.1
cm138a	6	8	4	55	4	268.3	55	4	216.5	19.3
cm150a	21	1	9	162	15	719.5	182	15	650.3	9.1
cmb	16	4	39	235	8	1641.2	244	8	1319.5	19.6
count	35	16	35	375	19	1979.6	375	19	1366.0	31.0
ex5	8	63	57	2391	12	9673.1	2426	12	8827.0	8.7
majority	5	1	5	36	5	250.9	36	5	184.8	26.3
mux	21	1	21	258	14	1343.1	258	14	881.4	34.4
pcle	19	9	36	257	10	1597.9	262	10	1394.0	12.8
t	5	2	3	24	3	143.9	25	4	135.4	5.9
tcon	17	8	17	117	4	816.2	117	4	721.2	11.6

Table 1: Combinational Benchmark Circuit with Latches Added

have also run our desensitization algorithm for large circuits successfully. The desensitizing functions remain simple for circuits with large number of gates and registers. For example, for the circuit `s5378` in benchmark suite `iscas89` which contains more than 1,000 gates and 162 registers, the most complicated desensitizing function has one inverter, one AND gate and one OR gate. Our algorithm is also quite fast. For example, the running time is less than 15 minutes on a SGI Indigo work station for circuit `s38584` which contains more than 13,000 gates and 1,400 registers.

6 Discussion and Future Work

In this section, we present two possible generalizations of the technique presented above and the future work it may lead to.

6.1 Controlling Values

Finding a controlling value for a general combinational element usually is not as easy as for a simple gate. However, for a general combinational element, for any given subset of the inputs, there exists a function h of these inputs such that when $h = 1$, the outputs is determined regardless of the values of the other inputs. For example, let $f(x_1, x_2, ..., x_n)$

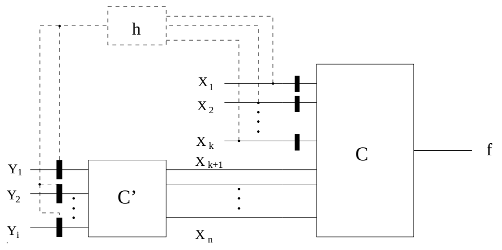

Figure 9: General combinational element

be the logic function that the combinational element C in Figure 9 realizes. If there is a register on each of the fanins $x_1, x_2, ..., x_k$, then we can find a function $h(x_1, x_2, ..., x_k)$ such that when $h = 1$, the value of $f(x_1, x_2, ..., x_n)$ is independent of the values of $x_{k+1}, x_{k+2}, ..., x_n$. Consequently, $h(x_1, x_2, ..., x_k)$ can be used to desensitize the registers at $Y_1, Y_2, ..., Y_i$, as shown in Figure 9. One possible choice for $h(x_1, x_2, ..., x_k)$ is:

$$h(x_1, x_2, ..., x_k) = \prod_{i=k+1}^{n} (f_{x_i} \bar{\oplus} f_{\bar{x_i}})$$

where f_{x_i} and $f_{\bar{x_i}}$ are the *shannon cofactors* of f with respect to x_i. After technology mapping, instead of simple logic gates, a sequential circuit contains combinational elements which are standard cells from the cell library. A breadth-first search algorithm similar to that in Section 4 can be used to determine the desensitizing signals based on the idea presented here. Because the sequential circuit is simpler (there are fewer standard cells than simple logic gates), computation of the desensitizing signals is also easier.

6.2 Timing Constraint

According to our discussion above, a register can be desensitized if its output is a *don't care* input to a combinational element in the subsequent clock cycle. In general, suppose the value of a signal A will determine the value of the output of a combinational element G in clock cycle $t_0 + k$. Then a register r can be desensitized by A if r is dominated by G and any path from r to G contains k registers (including r).

6.3 Future Work

Although the desensitizing functions we obtained for the benchmark circuits are all very simple, in general, the extra logic for generating the desensitizing signals will increase both the power consumption and the area of the circuit. However, if f is a desensitizing signal for a register r, then any signal g can be a desensitizing signal for r if $g = 1$ implies $f = 1$. Here, we are trading the opportunities for deactivating a portion of a circuit (using g instead of f as the desensitizing signal) for a simpler logic circuit to generate the desensitizing signal.

Our technique can also be combined with *retiming* transformation [7]. By moving the desensitized registers toward the primary inputs, more power reduction can be achieved since the larger the partial network between the desensitized registers and the desensitizing signal, the greater the potential power reduction will be. Retiming may also reduce the logic overhead by moving the registers toward the primary outputs. However, retiming with desensitized registers is an important topic that deserves further attention.

Circuit				Original			With DS			% Power
Name	I	O	R	Lits	Lvls	Power(μw)	Lits	Lvls	Power	Reduction
bbara	4	2	7	126	17	236.8	127	18	211.1	10.9
bbtas	2	2	5	61	6	249.2	61	6	214.7	13.8
beecount	3	4	6	100	7	331.7	103	8	304.0	8.4
dk16	2	3	15	464	11	1511.5	468	12	1395.3	7.7
dk17	2	3	4	115	9	439.4	116	10	407.1	7.4
dk512	1	3	8	113	13	632.9	114	114	599.2	5.3
ex1	9	19	11	360	13	937.0	363	13	814.5	13.1
ex3	2	2	7	143	9	642.2	143	9	602.2	6.2
ex4	6	9	10	159	8	853.0	166	9	788.9	7.5
ex6	5	8	9	185	9	726.8	187	10	652.1	10.3
lion9	2	1	3	47	7	146.3	48	8	136.0	7.0
lion	2	1	3	35	6	115.7	36	7	96.9	16.2
opus	5	6	6	144	10	515.8	145	10	472.5	8.4
styr	9	10	12	720	16	1230.1	723	16	1184.8	3.7
train11	2	1	7	95	6	320.1	96	6	284.6	11.1
s298	3	6	26	274	14	824.3	282	15	740.3	10.2
s344	9	11	17	268	15	644.1	269	15	557.5	13.4
s349	9	11	17	261	15	561.8	262	15	470.5	16.3
s444	3	6	27	326	18	586.7	334	19	470.6	19.8
s510	19	7	23	484	16	1385.6	490	17	1184.5	14.5

Table 2: Sequential Benchmark Circuits

References

[1] Miron Abramovici, Melvin A. Breuer, and Arthur D. Friedman. *Digital Systems Testing and Testable Design.* AT&T Bell Laboratories and W. H. Freeman and Company, 41 Madison Avenue, New York, NY 10010, 1990.

[2] Mazhar Alidina, Jose Monteiro, Srinivas Devadas, Abhijit Ghosh, and Marios Papaefthymiou. Precomupation-based sequential logic optimization for low power. In *International Conference on Computer-Aided Design*, pages 74–81, 1994.

[3] L. Benini, P. Siegel, and G. DeMicheli. Automatic synthesis of gated clocks for power reduction in sequential circuits. *IEEE Design and Test of Computers*, pages 32–40, December 1994.

[4] A. Chandrakasan, T. Sheng, and R.Brodersen. Low power cmos digital design. *IEEE Journal of Solid-state Circuits*, pages 473–484, April 1992.

[5] S. Devadas and S. Malik. A survey of optimization techniques targeting low power vlsi circuits. In *Design Automation Conference*, pages 242–247, 1995.

[6] A. Ghosh, S. Devadas, K. Keutzer, and J. White. Estimation of average switching activity in combinational and sequential circuits. In *Design Automation Conference*, pages 253–259, 1992.

[7] Charles E. Leiserson, Flavio M. Rose, and James B. Saxe. Optimizing synchronous circuitry by retiming (preliminary version). In *Advanced Research in VLSI and Parallel Systems*, pages 87–116, 1983.

[8] J. Monteiro, S. Devadas, and B. Lin. A methodology for efficient estimation of switching activity in sequential logic circuits. In *Design Automation Conference*, pages 12–17, 1994.

[9] Farid Najm. Transition density: A new measure of activity in digital circuits. *IEEE Transcations on Computer-aided Design of Integrated Circuits and Systems*, pages 310–323, February 1993.

[10] D. Pham and et al. A 3.0w 75specint92 85specfp92 superscalar risc microprocessor. In *IEEE International Solid-State Circuits Conference*, pages 212–213, 1994.

[11] H. Savoj, H. Touati, and R. K. Brayton. Extracting local don't cares for network optimization. In *International Conference on Computer-Aided Design*, pages 514–517, 1991.

[12] Ellen M. Sentovich and et al. Sis: A system for sequential circuit sysnthesis. Technical Report UCB/ERL M92/41, Department of Electrical Engineering and Computer Science, University of California, Berkeley, Electronics Research Laboratory, Berkeley, CA 94720, 1992.

[13] C-Y Tsui, M. Pedram, and A. Despain. Exact and approximate method for switching activity estimation in sequential logic circuits. In *Design Automation Conference*, pages 18–23, 1994.

[14] John F. Wakerly. *Digital Design Principles and Practices.* Prentice Hall, Englewood Cliffs, NJ 07632, 1990.