10-1-1993

Logic Synthesis for Reliability - An Early Start to Controlling Electromigration and Hot Carrier Effects

Kaushik Roy
Purdue University School of Electrical Engineering

Sharat C. Prasad
Texas Instruments

LOGIC SYNTHESIS – AN EARLY START TO CONTROLLING ELECTROMIGRATION AND HOT CARRIER EFFECTS

KAUSHIK ROY, PURDUE UNIVERSITY
SHARAT PRASAD, TEXAS INSTRUMENT

TR-EE 93-34
OCTOBER 1993

SCHOOL OF ELECTRICAL ENGINEERING
PURDUE UNIVERSITY
WEST LAFAYETTE, INDIANA 47907-1285

Logic Synthesis for Reliability – An Early Start to Controlling Electromigration and Hot Carrier Effects

Kaushik Roy
School of Electrical Engineering
Purdue University
West Lafayette, IN 47907-1285

Sharat C. Prasad
Integrated Systems Laboratory
Texas Instruments Inc., MS 446
Dallas, TX 75243

Please direct all correspondence to :
Kaushik Roy
Ph: 317-494-2361
Fax : 317-494-6440
kaushik@ecn.purdue.edu

Logic Synthesis for Reliability - An Early Start to Controlling Electromigration and Hot Carrier Effects

Abstract

Designing reliable CMOS chips involve careful circuit design with attention directed to some of the potential reliability problems such as *electromigration* and *hot carrier effects*. This paper considers logic synthesis to handle electromigration and hot carrier degradation early in the design. phase. The electromigration and the hot carier effects are estimated at the gate level using signal activity measure, which is the average number of transitions at circuit nodes. Logic can be optimally synthesized suited for different applications requiring different types of inputs for higher reliability and low silicon area. Results have been obtained for MCNC synthesis benchmark examples.

Category: Logic Synthesis (4.1)

1 Introduction

The *electromigration* and *hot* carrier induced circuit performance degradation is one of primary concerns for long-term reliability of MOS VLSI circuits. The conventional approach of post layout/fabrication reliability estimation is becoming increasingly costly due to lengthening of the design cycle due to previously unforseen reliability problems. A logical alternative is to be able to predict reliability at the logic or gate level so that circuits can be synthesized with reliability in mind. If stochastic properties of primary input signals are known, then signal activity [4] can be used to predict electromigration and hot carrier induced degradation. Signal activity is associated with average number of transitions occurring at any particular node of a circuit. Iyer et. al. [6] have shown that signal activity is also a measure of the stress that can cause failures in digital circuits.

The problem of determining when and how often transitions occur at a node in a digital circuit is difficult because they depend on the applied input vectors and the sequence in which they are applied. During the course of normal operation each of these vary widely. Therefore probabilistic techniques have been resorted to. All reported methods of estimating the probability of a transition at a circuit node involve estimation of signal probability, which is the probability that the logic value at a node is a ZERO or an ONE. Computing signal probabilities has attracted much research [8, 11, 9, 10]. In [8], a simple and general but inefficient scheme based on symbolic algebraic expressions is described. In [11], a relatively efficient algorithm to estimate the range (a sub-range of [0, 1]) that the signal probability of a node lies within is presented. But there exist doubts as to whether the ranges are narrow enough to be of use. The algorithm presented in [9] is very efficient but the values computed by it are approximate. A more sophisticated algorithm is proposed by Kapur and Mercer in [10]. This algorithm works by generating tighter bounds than [0, 1] to assign to the branch cut at the point of fanout. These bounds are computed by a prediction scheme. The accuracy of the results depends on the performance of the prediction scheme. There are classes of circuits for which the prediction scheme fails.

Combinational logic synthesis have been conventionally targeted to reducing area, critical path delay, and testability [18]. Recently, power dissipation during the logic synthesis process has been considered [3]. However, there exists no such tool which consider reliability early during the design process. Recently, Li and Hajj [17] have been able to use input reordering at the circuit level to eliminate the hot carrier effect degradation on performance. Sun, Leblebici, and Kang have used macro-models for evaluating hot-carrier related degradations of simple CMOS circuits [15]. In this paper we consider two reliability issues – electromigration and hot carrier effects – during the logic synthesis process. The reliability issues are estimated by modeling the primary inputs of a circuit as stochastic processes having a certain signal probability (probability that the input has a logic value of ONE) and signal activity, which measures the average number of transitions at circuit nodes. As the logic synthesis process is driven by the input signal probabilities and activities, a particular circuit can be optimally synthesized in different ways suited for different applications requiring different types of inputs. The synthesis process takes two level Boolean logic description and transforms it into multi-level form to minimize area and at the same time increase the reliability measures. The optimization process is iterative. During each iteration, the best sub-expression from among all promising common sub-expressions is selected. The objective function is based on both area and reliability measure.

The paper is organized as follows. Section 2 considers the preliminaries and the basic definitions required for understanding the signal activity based synthesis process. Section 2.2 and Section 3 consider signal probability calculation for circuits with reconvergent fanout. Section 4 considers electromigration and hot electron degradation and models those in terms of signal activity. Section 5 describes multilevel logic synthesis based on reliability measure. Results of our analyses on MCNC benchmark examples are given in Section 6. Section 7 summarizes the results and draws the conclusions.

2 Preliminaries and Definitions

2.1 Multilevel logic representation

Multilevel logic can be described by a set $\mathfrak{F}$ of completely specified Boolean functions. Each Boolean function $f \in \mathfrak{F}$, maps one or more input and intermediate signals to an output or a new intermediate signal. A circuit is represented as a Boolean network. Each node has a Boolean variable and a Boolean ezpression associated with it. There is a directed edge to a node g from a node f, if the expression associated with node g contains in it the variable associated with f in either true or complemented form. A circuit is also viewed as a set *of* gates. Each gate has one or more input pins and (generally) one output pin. Several pins are electrically tied together by a signal. Each signal connects to the output pin of exactly one gate, called the driver gate.

2.2 Signal Probability and Signal Activity

This subsection will briefly describe the concepts of signal probability and activity, the details of which are given in [4]. Let $g(t)$, t $\in (-\infty, +\infty)$, be a stochastic process which takes the values logical 0 or logical 1, transitioning from one to the other at random times. A stochastic process is said to be strict-sense stationary (SSS) if its statistical properties are invariant to a shift of the time origin. More importantly, the mean of such a process does not change with time. If a constant-mean process $g(t)$ has a finite variance and is such that $g(t)$ and g(t $+ \tau$) become uncorrelated as $\tau \rightarrow \infty$, then $g(t)$ is mean-ergodic. As in [4], we use the term *mean-ergodic* to refer to regular processes which are mean-ergodic and satisfy the two conditions of finite-variance and decaying auto-correlation. For such a signal we can define the following terms:

Definition: The signal probability of signal g is given by:

$$P(g) = \lim_{T \rightarrow \infty} \frac{1}{2T} \int_{-T}^{+T} g(t)dt$$

Definition: The signal activity of a logic signal $g(t)$ is given by

$$A(g) = \lim_{T+m} \frac{n_g(T)}{T}$$

where $n_g(t)$ is the number of transitions of $g(t)$ in the time interval between $\frac{-T}{2}$ and $\frac{+T}{2}$. If the primary inputs to the circuit are modeled as mutually independent *SSS* mean-ergodic 0-1 processes then the probability of signal $g(t)$ assuming the logic value ONE at any given time t becomes a constant independent of time and is referred to as the *equilibrium* signal probability of random quantity $g(t)$ and is denoted by $P(s = 1)$, which we refer to simply as signal *probability*. Secondly, $A(g)$ becomes the expected number of transitions per unit time.

Let us consider a multi-input, multi-output logic module M which implements a Boolean function. M can be a single logic gate or a higher level circuit block. We assume that the inputs to M, $g_1, g_2, ..g_n$ are mutually independent processes each having a signal probability of $P(g_i)$, and a signal activity of $A(g_i)$, $i \leq$ n. The signal probability at the output can be easily computed using one of the methods described in [12]. For example, if P_1, P_2, and P_3 are the input signal probabilities to a three input AND gate, the output signal probability is given by $P_1P_2P_3$, whereas, for an OR gate the output signal probability is $1-(1-P_1)(1-P_2)(1-P_3)$. For an inverter, the ouput signal probability is simply $(1-P_1)$, where P_1 is the input signal probability. The signal activity at any output h_j, of M is given by

$$A(h_j) = \sum_{i=1}^{n} P\left(\frac{\partial h_j}{\partial g_i}\right) D(x_i) \tag{1}$$

Here x_i, i = 1,.., n are the module inputs and $\partial h/\partial g$ is the boolean difference of function g with respect to h and is defined by

$$\frac{\partial h}{\partial g} = h\,|_{g=1} \oplus h\,|_{g=0} = h_g \oplus h_{\overline{g}} \tag{2}$$

Figure 1 shows the propagation of signal activity through AND, OR, and NOT gates. The signal probabilities and the circuit activities at the primary inputs to a circuit are assumed to be available.

The power dissipation in a CMOS circuit can be modeled in terms of the circuit activity of the nodes. In CMOS majority of the power is dissipated when there is a transition (ONE to ZERO or ZERO to ONE) at the ouput of a gate. The rest of the power is dissipated due

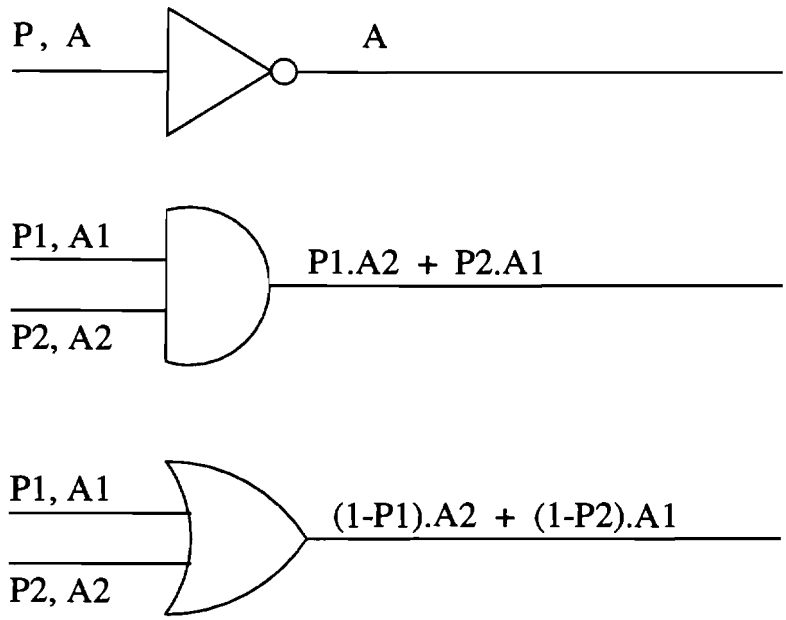

Figure 1: Propagation of circuit activities through basic gates

to direct-path short circuit current, and leakage current and are not dominant. Hence, the average power dissipation in a CMOS circuit can be written as [4, 3]:

$$POWER_{avg} = \frac{1}{2} V_{DD}^2 \sum C_i A(g_i) \tag{3}$$

summing over all circuit nodes g;. The capacitive load C_i at each node of a circuit can be approximated by the fanout factor at that node.

If the average current drawn from the supply voltage V_{dd} is I, then the: average power dissipated in the logic is $V_{dd}I$. Hence, it is clear from Equation (3) that the average supply current (or Vdd current) is given by

$$I = \frac{1}{2} V_{dd} \sum C_i A(g_i) \tag{4}$$

where the summation is taken over all circuit nodes g_i.

3 Accurate Signal Probability Calculation

The probability of a logic signal s expressed as $P(s = 1)$ or P_s is a real number in the range [0, 1] which expresses the probability that signal s has logic value 1. It is easy to compute signal probabilities for all nodes in a circuit that does not have reconvergent fanout. There

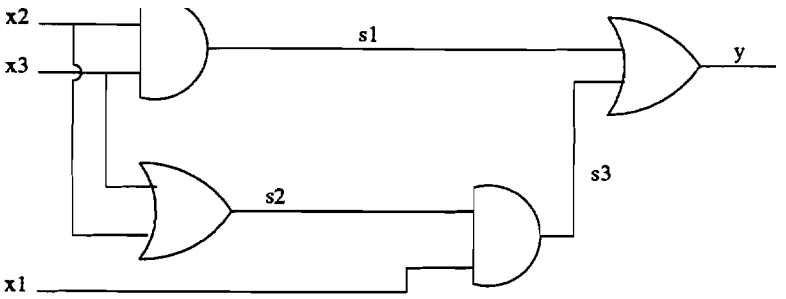

Figure 2: An example circuit with reconvergent fanout

has been much work on bounding or estimating the signal probabilities, but the results obtained are either inexact or approximate. Since having correct signal probabilities was important to the investigation of the current problem, it was decided to use the general algorithm proposed in [8] and do as efficient an implementation as possible. The algorithm is as follows.

Algorithm : Compute Signal Probabilities

Inputs : Circuit, signal probabilities of all the inputs

Output : Signal probabilities for all nodes of the circuit

Step 1 : For each input signal and gate output in the circuit, assign a unique variable.

Step 2 : Starting at the inputs and proceeding to the outputs, write the expression for the output of each gate as a function (using standard expressions for each gate type for probability of its output signal in term of its mutually independent input signals) of its input expressions.

Step 3 : Suppress all exponents in a given expression to obtain the correct probability expression for that signal.

Table 1 illustrates the computation of signal probabilities and circuit activities for a circuit implementing function $y = x_1 \wedge (x_2 \vee x_3) \vee x_2 \wedge x_3$ as shown in Figure 2.

We devised a data representation for signal probability expressions which is memory efficient and which allows us to perform the necessary operations efficiently. In this representation we have taken advantage of the fact that exponents have been suppressed and therefore, a signal probability expression may contain a variable (assigned to one of the in-

Table 1: Signal probabilities and circuit activities

Node	Signal Probability Expression	Signal Probability	Signal Activity
x_1	a	0.5	2.7
x_2	b	0.5	13.5
x_3	c	0.5	.3
s_1	bc	0.25	6.9
s_2	$b + c - bc$	0.75	6.9
s_3	ab + ac − abc	0.375	5.475
y	$ab + bc + ac - 2abc$	0.5	8.419

puts) raised to power 1 or may not contain it. So each product term may be regarded as a set with variables as its elements. The multiplication of two product terms can be achieved by taking the union of the corresponding sets. The primary inputs of the circuit under consideration are arbitrarily ordered and assigned indices. Let x_j, $0 < j < M$, be the primary input signals. Let p_j be the signal probability variable (a, b, and c in the example a'bove) assigned to input xj, i.e. $P(x_j = 1) = p_j$. A product term Q_i (a, bc, $-2abc$, etc. in the examples above) is represented as a pair (a_i, β_i), where both a_i and β_i are integers. α_i is called the *coefficient* of the term and may be negative or positive. β_i is regarded as a bit string. Bit j of β_i, written β_{ij}, is 1 if and only if the corresponding product term contains the variable p; and is 0 otherwise. Hence for the product term −2abc in the example above, coefficient = −2 and the bit array = 111. When two product terms Q_i and Q_j are multiplied, the resulting product term Q_k is given by (α_k, β_k), where $\alpha_k = a_i * \alpha_j$, and $\beta_{kl} = \beta_{il} \wedge \beta_{jl}$. It is easy to see that we can define a full order relation on the set of all possible product terms. $Q_i < Q_j$ if $\beta_i < \beta_j$, where both β_i and β_j are interpreted as integers. Each probability expression is represented as an ordered list of its product terms, i.e., $P(G) = (Q_1, Q_2, .., Q_{n_G})$. It is obvious that the sum of two expressions $P(G)$ and $P(H)$ can be determined in $O(n_G + n_H)$ time and the product in $O(n_G * n_H)$ time.

In the preceding discussion it was implicitly assumed that the word size of the machine is larger than M. When this is not the case multiple words may be used to implement each

β_i.

We have assumed that the primary input signals $(i_1, ...i_m)$ are mutually independent. If they are not, then we can find a set of n mutually independent signals $(i'_1, ..., i'_n)$, n $\leq$ m, such that each i_j can be expressed in terms of i'_k. Now, the signal probabilities of i_js will be given by symbolic expressions containing signal probabilities of i'_ks. The signal probability expressions for internal nodes will also be in terms of signal probabilities of i'_ks rather than i_js.

Electromigration and Hot Carrier Degradation

Electromigration is a major reliability problem caused by transport of atoms; in a metal line due to the electron flow and it affects both MOS and bipolar components. 'Thermally activated ions of the conductor which normally self-diffuse in all directions are given a direction of net motion due to momentum transfer from the conducting electrons. Hence, the ions move "downstream" with the electrons [16]. A divergence of the ion flux ultimately gives rise to a circuit failure. A positive divergence leads to an accumulation of vacancies to form a void in the metal and ultimately an open circuit. A negative divergence leads to a buildup of metal, called a hillock, which can ultimately lead to a short circuit to adjacent or overlying metal. Electromigration is a wearout mechanism and is caused by persistent current stress. The time-to-failure is a lognormally distributed random variable. It is characterized by the Mean-Time-to-Failure (MTF), which depends on the current density in the metal line. Some recent models [13] of MTF predict that, at least under pulsed-DC conditions, the average current is sufficient to predict MTF as follows:

$$MTF = \frac{A}{\mathbf{J}^2} \tag{5}$$

where A has a statistical distribution and is independent of J, and J is the ,average current density. Hence, $MTF \propto J^{-2}$. The dependence of MTF on current density at least represent a first order approximation of current stress in a wire. Equation (4) represents the average current through a gate in terms of signal activity. In turn current density is proportional to

the average current. Hence, signal activity can can be used to model *electromigration effect* E to the first order of approximation. E for a gate is minimized if $\mathbf{1}^2$ is minimized or if $\boldsymbol{I}$ is minimized. Hence, from Equation (4), E can be minimized if the following expression is minimized:

$$\frac{1}{2}V_{dd}\sum C_i A(g_i)$$

As V_{dd} is constant over all the gates of the circuit, the electromigration effect can be represented as:

$$E = \sum C_i A(g_i) \tag{6}$$

where the summation is taken over all the gates.

As MOSFET devices are scaled down to small dimensions, certain physical mechanisms start to cause degradation in the device parameters, causing major reliability problems. One such mechanism is the injection of *hot carriers* into the MOS gate oxide layer [16]. Trapping of these carriers in the gate insulator layer causes degradation in the transistor transconductance and threshold voltage. Moreover, because oxide charging is cummulative over time, the phenomenon limit the useful *life* of a device. Hot carriers are electrons and holes in. the channel and pinch-off regions of a transistor which have gained so much energy from lateral electric field produced by the source-drain voltage that their energy distribution is much greater than would be predicted if they were in thermal equilibrium with the lattice. The higher mobility of the electrons in silicon makes the impact worse in n-channel transistors. After the electrons have gained about 1.5 eV of energy, then can lose it via impact ionization (electron-hole pair creation) which, in turn, gives rise to hole substrate current. It is widely accepted that the MOSFET substrate current is a good indicator of the severity of the degradation. The average substrate current is given by [17]

$$I_{sub} = \frac{C_i}{B_i} I_{ds}(V_{ds} - V_{dsat}) exp\{\frac{-B_i l_c}{V_{ds} - V_{dsat}}\}$$

where C_i, B_i are parameters independent of current, and I_{ds} and V_{ds} respectively represent the drain-source current and voltage. The magnitude of I_{sub} is dependent on the switching

activity of EAOS transistors and the duration a transistor stays in the saturation region. Since hot carrier degradation occurs when transistor is in saturation region, which in CMOS circuits, happens only during transitions, it follows that higher the signal activity at the gate ouput, the more damage a transistor experiences. The duration a transistor stays in the saturation region is dependent on input slew rate and the load capacitance of a gate. The load capacitance is in turn dependent on the fanout of the gate. Hence, the first order approximation of hot electron degradation at the gate level (H_{gate}) can be modeled in terms of signal activity of a gate output and its fanout factor as follows: .

$$H_{gate} = A_{gate} \cdot f_{gate}$$

where A_{gate} and f_{gate} represent the signal activity and fanout of a gate.

One can write the ***age*** of a transistor that has been operating for time T as follows [13]:

$$Age(T) = \int_{\frac{+T}{2}}^{\frac{-T}{2}} \frac{Ids}{WH} \left(\frac{I_{sub}}{I_{ds}} \right)^{m} \tag{7}$$

where I_{ds} and I_{sub} are the MOSFET drain to source and substrate currents respectively. W is the channel width, and H and m are parameters independent of transistor currents.

It can be observed from the equations for E and H_{gate} that for designing hot carrier and electromigration resistant circuits at the gate level, the following expression has to be minimized:

$$\sum A_{gate} \cdot f_{gate} \tag{8}$$

summing over all the gates of the circuit. The above expression is given by R and can be thought of as an ***unreliability factor.*** The larger is the value of R, the larger will be the unreliability of the circuit.

5 Logic Synthesis for Reliability

A complicated two level Boolean function is often implemented to have additional levels between inputs and outputs. Multilevel optimization of a set of such Boolean functions involves creating new intermediate signals and/or removing some of the existing ones to

achieve reduction in area and to meet other design constraints such as performance. The global area optimization process of MIS [18] turned out to be very well-suitedl for extensions to consider the impact on reliability of a circuit. The input to the optimization process is a set of Boolean functions. *A* procedure, called *kernel* finds some or all *cube-free* multiple or single cube divisors of each of the functions and retains those divisors which are common to two or more functions. The best few common divisors are factored out and the affected functions are simplified by a second procedure called *substitution.* This process is repeated until no common divisors can be found. The "good"ness of a divisor is measured by the magnitude of area saving it brings about. In our system, this has been extended to take reliability also into account.

5.1 Factorization

In this section we consider the effect of factoring out a common subexpression from several expressions and also that of selecting a factor from amongst possible factors. Let g = $g(u_1, u_2, .., u_K)$; K $\geq$ 1, be a common subexpression of functions f_1, f_2, .., f_L; L $\geq$ 2. Let v_1, v_2, .., v_M; $M \geq 0$ be the nodes internal to g. Each input u_k to g is either primary input or the output of a node in the circuit. Figure **3** is a pictorial representation of the circuit. When g is factored out of f_1, f_2, .., f_L, the signal probabilities and circuit activities at all the nodes of the Boolean network remain unchanged. However, the capacitances at the output of the driver gates of $u_1, u_2, .., u_K$ change. Each such gate now drives (L $-$ 1) fewer gates than before. This results in a reduction in the *unreliability factor* which is given by

$$(L-1)C_0 \sum_{k=1}^{K} n_{u_k} A(u_k)$$

Here $A(x)$ is the signal activity at node x, n_{u_k} is the number of gates belonging to node g and driven by signal u_k (there are gates not belonging to g which are also driven by u_k) and C_0 is the load capacitance due to a fanout equal to one gate.

The driver gate of the newly created node g drives exactly as many gates as the driver gates of all its copies (which existed prior to factorization) taken together, so there is no

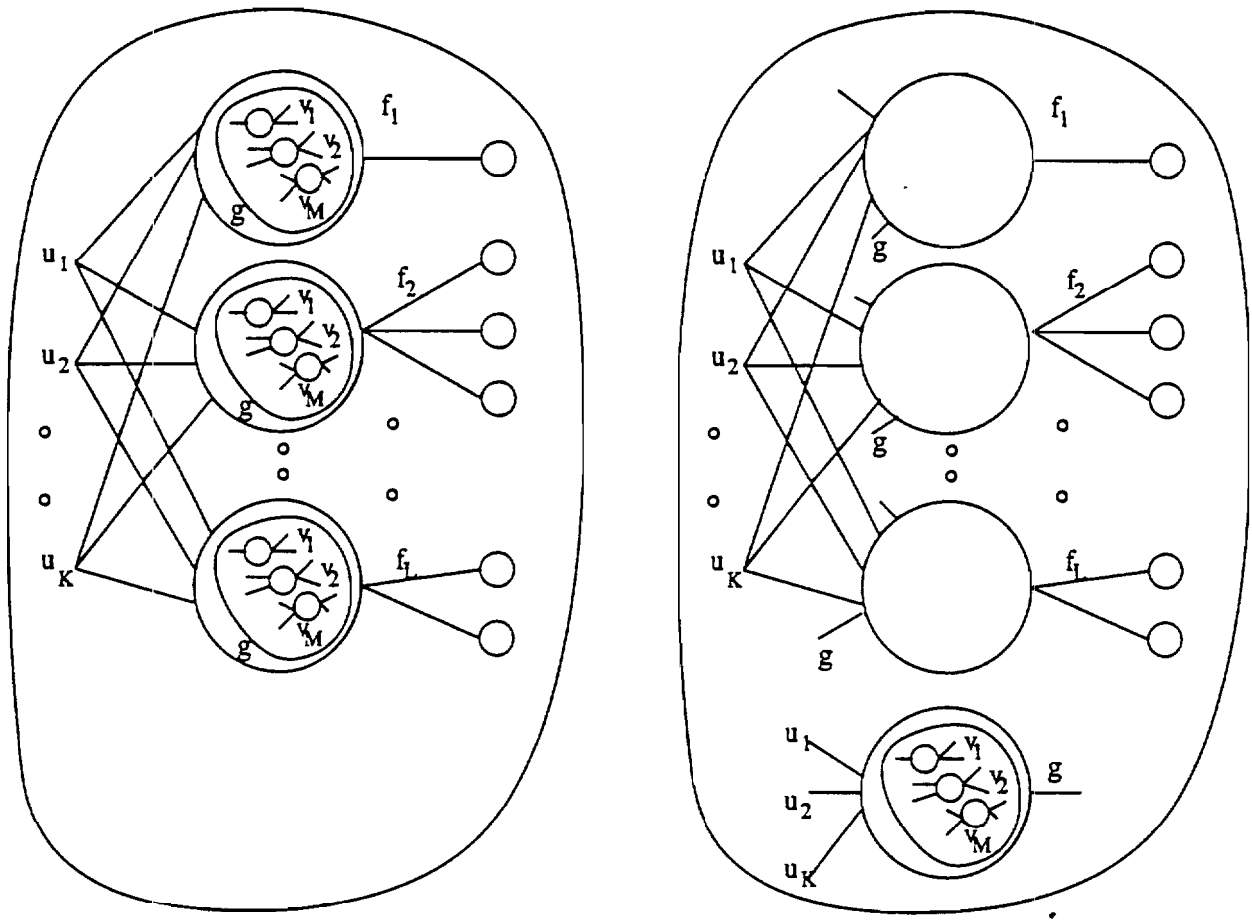

Figure **3:** Factoring a common subexpression

change in this component of reliability. Since there is only one copy of g in place of L, there are L − 1 fewer copies of internal nodes v_1, v_2, .., v_M switching. The reliability improvement is given by

$$(L-1)C_0 \sum_{m=1}^{M} n_{v_m} A(v_m)$$

Here n_{v_m} is the number of gates driven by signal v_m.

The total reliability improvement on factoring out g is the sum the above two components and is given by

$$\Delta R(g) = (L-1)C_0 \left(\sum_{k=1}^{K} n_{u_k} A(u_k) + \sum_{m=1}^{M} n_{v_m} A(v_m) \right) \tag{9}$$

The area saving $\Delta A(g)$ due to a divisor g is found as in [18]. Let $T(g)$ be the number of literals in the factored form of g. Then,

$$\Delta A(g) = (L-1)(T(g)-1) - 1$$

The net improvement is given by,

$$S(g) = \alpha_R * \frac{\Delta R(g)}{R_T} + \alpha_A * \frac{\Delta A(g)}{A_T}. \tag{10}$$

Here R_T and A_T are the area and reliability factor of the input Boolean network, and α_R and α_A are weight factors: $0 \leq \alpha_A, \alpha_R \leq 1$, and $\alpha_R + \alpha_A = 1.0$.

5.2 The optimization procedure

At the beginning of the optimization procedure, signal probabilities and activities for each internal and output node is computed. Each time a common divisor g = $g(u_1, u_2, .., u_K)$ is factored out, the $P(u_k = 1)$ and $A(u_k)$, 1 ≤ k ≤ K, are known but $P(v_m = 1)$ and $A(v_m)$, 1 ≤ m ≤ M, are not. The latter are computed when $\Delta R(g)$ is being evaluated and are retained. Thus once again $P(s = 1)$ and $A(s)$, for each node s are known.

The parameter N_0 is used to control the number of kernel intersections (cube free divisors common to two or more functions) which are substituted into all the functions before the set of kernel intersections is recomputed. Recomputing after a single substitution is wasteful as

only some of the functions have changed. On the other hand, with each substitution, some of the kernel intersections become invalid.

Algorithm : Reliability driven multilevel logic optimization

Inputs : Boolean network F, input signal probability $P(x_i = 1)$ and activity $A(x_i)$ for each primary input x_i, N_0

Output : Optimized Boolean network F', $P(s = 1)$ and $A(s)$ for each node in the optimized network.

Step *0* : Compute $P(s = 1)$ and $A(s)$ for each node s in F.

Step 1 : Repeat steps 2 through 4.

Step 2 : $G' = \bigcup_{f \in F} K(\mathrm{f})$, where $K(\mathrm{f})$ = set of all divisors of f. Set of kernels (cube free divisors) is computed for each function. G' is the union of all the sets of kernels.

Step *3* : $G = \{\mathrm{g} \mid (\mathrm{g} \in G') \wedge (\mathrm{g} \in K(f_i)) \wedge (\mathrm{g} \in K(f_j)) \wedge (\mathrm{i} \neq \mathrm{j})\}$. G, the set of kernel intersections, is the set of those kernels which apply to more than one function.

Step 4 : Repeat steps 5 through 7 N_0 times

Step 5 : Find g, p_g, d_g such that

$$(g \in G) \wedge (\forall h \in G)[S(g) \geq S(h)] \wedge (p_g = P(g = 1)) \wedge (d_g = A(g))$$

If $\Delta A(g) < 0$, terminate procedure. g is the kernel intersection which brings about largest saving. The signal probability and activity of the output signal of g are remembered. If the area component of total saving is negative, there are no more multiple-cube divisors common to two or more functions and so we stop.

Step *6* : For all f such that $\mathrm{f} \in F \wedge \mathrm{g} \in K(f)$, substitute variable g in f in place of the subexpression $g(u_1, u_2, .., u_K)$. Each function, which has the expression g as one of its kernels, has the new variable g substituted into it in place of the expression.

Step 7: $F' = \mathrm{F} \bigcup \{g\}$, G = G − {g}. P(g = 1) = p_g, $A(g) = d_g$. New function g is added to the set of functions F. The newly added node is assigned signal probability and activity values from step 5.

Table 2: Experimental results

Circuit	*Inputs*	*Unoptimized*		$\alpha_R = 0,\ \alpha_A = 1$		$\alpha_R = 1,\ \alpha_A = 0$	
		Area	*Unrel*	*Area*	*Unrel*	*Area*	*Unrel*
duke2	22	959	2104	702	964	798	851
vg2	25	390	880	322	528	349	481
misex2	25	164	168	163	162	164	159
rd84	25	1107	2540	831	1851	903	1522
apex4	9	5714	12008	4981	7723	5432	6815
miscex3c	14	5431	15100	4403	9914	4844	9006

6 Implementation and Results

The algorithms for logic synthesis to achieve higher reliability have been implemented in LISP on an Explorer workstation. Table 2 shows the results for MCNC combinational benchmark examples. In each case each input signal was assigned a signal probability of 0.5 and a signal activity of a randomly generated number (once for every example and then the same value for all subsequent runs of that example) in the range from .O1 to 50 million transitions per second. It should be noted that circuits would be synthesized differently if a different signal probabilities and activities were used. The area is in terms of literals and the unreliability factor R represents the reliability of the circuit.

$(\alpha_A = 1,\ \alpha_R = 0)$ corresponds to the traditional multilevel optimization where reliability optimization is not considered. At first the choice of parameters $(\alpha_A = 0,\ \alpha_R = 1)$ may appear strange, and one may expect the resulting areas to be very large. But as the results show that is not what happens. Optimization of reliability is achieved by eliminating, at a time, redundant copies of one subexpression which has higher weighted sum (weighted by the capacitance at the node) of signal activities at its nodes than other subexpressions. Elimination of any common subexpression automatically results in reduction in area.

7 Coriclusions

A logic synthesis system has been developed which considers reliability degradation due to electromigrittion and hote carrier effects early in the design phase. Electromigration and hot carrier effects can be modeled in terms of signal activity, and hence, a particular circuit can be synthesized in different ways for different applications which require different types of inputs. Results on MCNC synthesis benchmarks show that a minimum area circuit is usually not associated with highest reliability.

8 Acknowledgement

The authors would like to thank Farid Najm for many fruitful discussions and Bhanu Kapoor for letting us to use the Boolean expression factorization utilities he developed.

References

[1] C. Mead, L. Conway, Introduction to VLSI systems, Addison-Wesley, Menlo Park, CA, 1980.

[2] H.B. Bakoglu, Circuits, Interconnections, and Packaging for VLSI, Addison-Wesley, Menlo Park, CA, 1990.

[3] K. Roy and S. Prasad, "Circuit Activity Based Logic Synthesis for Low Power Operations," IEEE Trans. on VLSI Systems, to appear in December 1993.

[4] F.N. Najm, "Transition Density, A Stochastic Measure of Activity in Digital Circuits," A*CM/IEEE* Design Automation Conf., 1991.

[5] A.Ghosh, S.Devdas, K.Keutzer, and J.White, "Estimation of Average Switching Activity in Combinational and Sequential Circuits," *ACM/IEEE* Design Automation Conf., 1992.

[6] R. Iyer, D. Rossetti, and M. Hsueh, "Measurement and Modeling of Computer Reliability as Affected by System Activity", ACM Trans. on Computer Systems, Vol. 4, No. 3, August, 1986, pp. 214-237.

[7] T.Lengauer, and K. Melhorn, "On the Complexity of VLSI Computations ", Proceedings of *CMU* Conference on VLSI, Computer Science Press, October 1981, pp. 89-99.

[8] K. P. Parker and E. J. McCluskey, "Probabilistic Treatment of General Combinatorial Networks," IEEE Trans. on Computers., vol. C-24, pp. 668 – 670, June 1975.

[9] B. Krishnamurthy and I. G. Tollis, "Improved Techniques for Estimating Signal Probabilities", IEEE Trans. on Computers., vol. C-38, pp. 1245 - 1251, July 1989.

[10] R. Kapur, and M. R. Mercer, "Bounding Signal Probabilities for Testability Measurement Using Conditional Syndromes", SRC Pub C92152, March 1992.

[11] J. Savir, G. Ditlow, and P. Bardell, "Random Pattern Testability", IEEE Trans. on Computers, January, 1984, pp 79-90.

[12] S. Ercolani, M. Favalli, M. Damiani, P. Olivio, and B. Ricco, "Estimatation of signal Probability in Combinational Logic Network", 1989 European Test Conference, pp. 132-138.

[13] C. Hu, "Reliability Issues of MOS and Bipolar ICs," Intl. Conf. on Computer Design, 1989, pp. 438-442.

[14] C. Hu et. al., "Hot-Electron-Induced MOSFET Degradation – Model, Monitor, and Improvement," IEEE Trans. Electron Devices, February, 1985.

[15] Y. Leblebici, W. Sun, and S. Kang, "Design-For-Reliability Rules for Hot-Carrier Resistant CMOS VLSI Circuits," *Intl. Symp.* Circuits and Systems, 1992, pp. 1254-1257.

[16] M. Woods, "MOS VLSI Reliability and Yield Trends," Proc. of IEEE, Dec. 1986, pp. 1715-1729.

[17] P. Li and I. Hajj, "Computer-Aided Redesign of VLSI Circuits for Hot-Carrier Reliability," *Intl. Conf.* on Computer Design, 1993 (to appear).

[18] R. Brayton, R. Rudell, A. Sangiovanni-Vincentelli, and A. Wang, "MIS: A Multiple-Level Logic Optimization System", IEEE Trans. on Computer-Aided Design, November, 1987, pp. 1062-1081.