

Optimal Assignment of High Threshold Voltage for Synthesizing Dual Threshold CMOS Circuits

Nikhil Tripathi, Amit Bhosle, Debasis Samanta, and Ajit Pal

Computer Science and Engineering Department

Indian Institute of Technology, Kharagpur

India-721302

Email: apal@cse.iitkgp.ernet.in

Abstract

Development of the process technology for dual threshold (dual V_{th}) CMOS circuit has opened up the possibility of using it to reduce static power in low voltage high performance circuits. It has been demonstrated that by using transistors of a low threshold voltage for gates on the critical path, and by using a high threshold voltage for gates in the off-critical path it is possible to significantly reduce leakage power consumption of a circuit without performance degradation. In this paper we have proposed a new algorithm to realize dual threshold CMOS circuits. Our algorithm produces significantly better results for the ISCAS benchmark circuits compared to the reported results.

1.0 Introduction

As dynamic power depends quadratically on the supply voltage V_{dd} and static power is directly proportional to it, supply voltage reduction has the most profound effect on power consumption of CMOS circuits. However, reduction in V_{dd} leads to an increase in delay, which results in performance degradation of the circuit. Scaling down the threshold voltage V_{th} by the same factor as V_{dd} is considered to be a solution to keep the same performance level. Unfortunately, reducing V_{th} in smaller geometry MOSFETs results in an exponential increase in the static power. Main component of the static power is the subthreshold leakage current. It has been reported that the power dissipation due to this component dominates the dynamic power at low threshold voltages [1]. In portable battery-driven systems, where large portion of circuits remain in standby mode for a long duration, this static power turns out to be a major issue. To alleviate this problem, the use of multiple threshold voltage CMOS (MTCMOS) circuits has been proposed as a viable solution. Technology for dual- V_{th} CMOS process has been developed making implementation of dual threshold CMOS circuits a reality [8]. Two basic strategies have

been proposed for the reduction of standby power using this dual V_{th} technology. In one, a high-threshold sleep transistor is inserted in series with low-threshold transistor stacks. The sleep transistor is turned off in the standby mode, leading to reduction in standby current. As this strategy suffers from several limitations [8], it is not a preferred approach. The other strategy uses low-threshold transistors for gates on the critical path and high threshold transistors on the off-critical path [2], [9]. In [2], a heuristic-based algorithm was presented which, for a given circuit with a fixed supply voltage (1V) and a fixed low threshold voltage V_{th}^L (0.2V) for gates on the critical path, finds a “static power optimum” high threshold voltage V_{th}^H , and a subset of the gates in the off-critical path which can be switched to V_{th}^H . This approach demonstrated significant savings in leakage power for the optimized circuits without degradation in performance. However, the algorithm in [2] employed simple backward breadth-first search (BFS) strategy to identify the subset of gates that can be switched to V_{th}^H . So far as the computational effort is concerned, the said algorithm is fast but it has two inherent drawbacks:

- i) It selects fewer gates for assigning V_{th}^H in the off-critical path; in fact more gates can be assigned leading to more savings in the leakage power.
- ii) After the assignment of V_{th}^H to a gate, the critical path may change. This dynamic change in critical path has not been taken into consideration.

Same problem has been addressed in [9], where a “near-optimal approach” has been proposed to obtain further improvements in power saving, in the range of 0% to 29.45%, compared to [2].

In this paper, we have also addressed the same problem and proposed a new algorithm to

overcome the limitations mentioned above. As anticipated, we have obtained significant improvement in results over [2] and [9] for the ISCAS benchmark circuits.

In Sec. 2 we provide the preliminaries covering the notations used in the subsequent sections. The models for measuring delay and standby leakage power is also presented in this section. In Sec. 3 details of the algorithm is presented. Implementation of the algorithm is described in Sec. 4. Experimental results are furnished in Sec. 5.

2.0 Preliminaries

A combinational circuit is represented by a directed acyclic graph $G(V, E)$. Each node (except the primary inputs and primary outputs) in G corresponds to a logic gate in the circuit and each edge maps to a connection in the circuit. The general form of the dual V_{th} optimization problem is to assign one of the two threshold voltages V_{th}^L and V_{th}^H to all the transistors of each gate such that some cost function is optimized subject to some constraints. The assigned threshold voltages to different gates are represented by labeling each node $v \in V$ (to which the transistors belong) by a variable x_i where $x_i = 0$ ($x_i = 1$) means the transistors in v_i are with $V_{th} = V_{th}^H$ ($V_{th} = V_{th}^L$).

The dual selection problem can be viewed in one of the two ways: either delay can be optimized subject to constraints on power or vice versa. Our problem is to reduce the standby leakage power subject to the constraint of not increasing the delay. In order to solve this problem, we need a model to estimate delay and standby leakage power. We have used the same models as used in [2]. These two models are summarized in the following two subsections.

2.1 Delay Estimation

The *propagation delay* through a node x , denoted as $t_p(x)$, defines how quickly the outputs responds to a change in the input. The propagation delay of a path Π , denoted as $pd(\Pi)$, is the sum of the propagation delays $t_p(i)$ of each node i along this path Π . It can be expressed as

$$pd(\Pi) = \sum_{\forall i \text{ in } \Pi} t_p(i) \quad (1)$$

The *arrival time* $T_a(x)$ is the propagation delay of each fan-in path of node x . among all the fan-in paths, there exists a path (or paths) which has a maximum propagation delay value $T_{\max}(x)$, where

$$T_{\max}(x) = \max\{T_a(x)[i]\} \mid i \in \text{all fanin} \quad (2)$$

The departure time $T_i(x)$ of node x is defined as

$$T_i(x) = T_{\max(x)}(x) + t_p(x) \quad (3)$$

The path, which determines the maximum speed of the circuit, is called the critical path. It can be noted that, there may be more than one critical path. *Critical delay* (T_{critical}) is the delay along the critical path. In [3], a model has been developed for estimating the delay of CMOS VLSI gates based on the commonly used *Elmore delay model* [7]. According to this model, the worst case propagation delay of a CMOS gate is given by

$$t_p = (t_{\text{PHL}} + t_{\text{PLH}}) / 2 \quad (4)$$

where,

$$t_{\text{PHL}} = 0.69 \sum_{j=1}^n \left(C_j \sum_{k=1}^j R_k \right) \quad (5)$$

for a NAND gate.

Here, the capacitance of each internal node j ($j = 1, 2, \dots, n$) in the n -input NAND gate is given as follows:

$$C_j = 2 \cdot C_{\text{dN}} \quad (6)$$

C_{dN} being the diffusion capacitance of an nMOS transistor and R_k being the on resistance of an nMOS transistor. Equation (5) gives the worst case delay when all the C_j 's in the pull down network discharged simultaneously. For the pull up network, the worst case occurs when only one pMOS transistor is "on". In this case, the worst case delay, let it be denoted by t_{PLH} , is given as

$$t_{\text{PLH}} = 0.69 R_p C_n \quad (7)$$

where, C_n is the capacitance of the gate at the output, which can be expressed as

$$C_n = F_O(C_{\text{gp}} + C_{\text{gN}}) + F_I C_{\text{dp}} + C_{\text{dN}} + F_O C_{\text{int}} \quad (8)$$

Here, F_O and F_I are the number of fan-out and fan-in of the gates, respectively. C_{gp} and C_{gN} are the gate capacitances of pMOS and nMOS transistors, respectively. C_{dp} and C_{dN} are the diffusion capacitances of pMOS and nMOS transistors, respectively. And C_{int} is the interconnect capacitance per fan-out.

In equation (7), R_p denotes the on-resistance of a pMOS transistor, which can be approximated as

$$R_p = \frac{R_{\text{pMOS}} \Big|_{V_{\text{out}}=V_{\text{dd}}} + R_{\text{pMOS}} \Big|_{V_{\text{out}}=V_{\text{dd}}/2}}{2} \\ = \frac{1}{2} \left(\frac{V_{\text{DS}}}{I_{\text{D}}} \Big|_{V_{\text{out}}=V_{\text{dd}}} + \frac{V_{\text{DS}}}{I_{\text{D}}} \Big|_{V_{\text{out}}=V_{\text{dd}}/2} \right)$$

$$\frac{V_{dd}}{k_p (V_{dd} - V_{thp})^\alpha} + \frac{V_{dd}}{k_p (2V_{dd} - V_{thp}) V_{dd} - \frac{V_{dd}^2}{2}} \quad (9)$$

Here, V_{thp} is the threshold voltage of pMOS, k_p is the gain factor of a pMOS and α is 2 and 1.3 for long channel and short channel MOSFETs, respectively.

Similarly, for an nMOS, the on-resistance (R_n) can be evaluated. For the sake of simplicity, we assume that $|V_{thp}| = |V_{thn}| = |V_{th}|$ and $R_n = R_p$. Using the above model the worst case propagation delay can be obtained.

2.2 Standby Leakage Power Estimation

Assuming the BSIM2 model [5], the subthreshold current of a MOS transistor is approximated as:

$$I_{sub} = A e^{\frac{q}{n'kT}(V_G - V_S - V_{th0} - \delta'V_S + \eta V_{DS})} \left(1 - e^{-\frac{qV_{DS}}{kT}} \right) \quad (10)$$

where, n' = subthreshold swing coefficient of the transistor

V_{th0} = zero bias threshold voltage

δ' = body effect coefficient

η = drain induced barrier lowering (DIBL) coefficient.

$$A = \mu_0 C_{ox} \frac{W}{L_{eff}} \left(\frac{kT}{q} \right)^2 e^{1.8}$$

here, μ_0 = zero bias mobility, C_{ox} = gate oxide capacitance per unit area.

The standby leakage power of a logic circuit can be expressed as follows:

$$P_{stdby} = \left(\sum_i I_{stdby_i} \right) \cdot V_{dd} \quad (11)$$

where, I_{stdby_i} = standby leakage current through each node i .

It may be noted that I_{stdby} depends on the gate topology as well as the input signal levels. The leakage current, in fact, refers to the current that flows after all charge stored in "isolated" internal nodes has been discharged and therefore the magnitude of this current can be determined completely by the input signal levels. An internal node is isolated if there is no path either to V_{dd} or GND.

Let us consider a static CMOS gate and assume that the transistors those are "on" are short circuits. Under this condition, if output is a

"1" then the leakage current is determined by the nMOS transistors in the pull down network that are "off". On the other hand, if the output is a "0" then the leakage current is determined by the pMOS transistors in the pull up network those are "off". As a particular example, let us consider a CMOS NAND gate with n inputs. If all the inputs are "1", the pull down network is shorted and the leakage current is determined by the pull up network. We can get the source and drain voltages of each transistor easily. Using equation (9), the leakage contribution of each transistor in the pull up network can be calculated separately. If at least one input is "0", the pull up network is shorted and standby leakage current is the leakage current through the pull down network. Suppose, there are n transistors, which are turned "off" in pull down network, the quiescent subthreshold leakage current through each of them must be identical. By equating the leakage currents of the transistors in the stack, V_S and V_{DS} of each transistor can be obtained. If $n = 1$, $V_{DS1} = V_{dd}$, $V_{S1} = 0$. Otherwise, the following equation can be used:

$$V_{DS2} = \frac{n'kT}{q(1 + 2\eta + \gamma')} \ln \left(\frac{A_1}{A_2} e^{\frac{qnV_{dd}}{n'kT}} + 1 \right) \quad (12)$$

$$V_{DSi} = \frac{n'kT}{q(1 + \gamma')} \ln \left(1 + \frac{A_{i-1}}{A_i} \left(1 - e^{-\frac{q}{kT}V_{DS_{i-1}}} \right) \right) \quad (13)$$

$2 < I \leq n$

$$V_{S_i} = \sum_{j=i+1}^n V_{DS_j} \quad 1 \leq I < n \quad (14)$$

$$V_{DS1} = V_{dd} - V_{S1} \quad (15)$$

where, $i=1$ represent the top transistor and $i = n$ represents the bottom transistor in the stack. Now, equation (10) can be used to calculate the quiescent leakage for any transistor in the stack, which is the leakage current of the pull down network. The average leakage current power of a circuit can be evaluated with random pattern applied to the primary inputs.

3.0 Algorithm

Our algorithm consists of three basic steps:

1. Initialization
2. Dual assignment of V_{th}
3. Computation for optimal V_{th}^H

Details of each of the above steps are given in the following subsections.

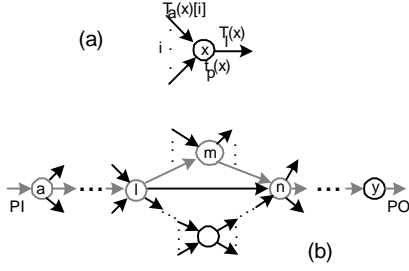


Fig. 1 During $Initialization(V_{th})$.

3.1 Initialization

We have assumed the directed acyclic graph representation $G(V, E)$ of a gate-level combinational circuit. This graph is then labeled in order to assign a number to each node to indicate the depth of the node in the graph. The level of each primary input is defined to be 0. The level of any node x , denoted as $l(x)$ can be calculated as follows:

$$l(x) = 1 + \max \{l(j)\} \quad (16)$$

where j varies for all fan-in nodes of node x .

For each primary input x , $t_p(x) = 0$, $T_a(x) = 0$, $T_l(x) = 0$, and $T_{max}(x) = 0$. For other nodes in the graph, $T_{max}(x)$ and $T_l(x)$ can be computed using equation (2) and (3), respectively. Figure 1(a) represents a typical node along with various parameters associated with it. By checking all the primary outputs and then backtracking, the critical delay and critical path(s) can be identified using a first-in-fast-out (FIFO) queue Q . A typical view of a part of the graph during initialization is shown in Fig. 1(b). The algorithm for the initialization $Initialization(V_{th})$ is given below:

Initialization(V_{th})

- Assign the V_{th} to each node x
- Calculate $t_p(x)$ of each node x
- Calculate $T_{max}(x)$ and $T_l(x)$ for each node x
- Calculate $T_{critical} = \max \{T_l(j) \text{ for all } j \in PO\}$

We have initialized the circuit by assigning a high threshold voltage (V_{th}^H) to all the gates of the circuit, i.e. it essentially configures the circuit to give minimum power. It may be noted that, the circuit is initialized by assigning low threshold voltage (V_{th}^L) to each node in [2] and [9] to configure the circuit for minimum delay.

3.2 Dual assignment of V_{th}

The next step is to select the nodes, which can be assigned the low threshold voltage in order to

decrease the delay of the circuit. We propose to assign low threshold voltage V_{th}^L to the nodes on critical path. Algorithm $DualAssignment(V_{thi})$ as given below performs this operation.

DualAssignment(V_{thi})

Repeat

Calculate $T_{critical} = \max \{T_l(j)\}$ for all $j \in PO$

For each j where all $j \in PO$

If $T_l(j) = T_{critical}$

Enqueue(Q, j)

While not Empty(Q)

$i = \text{Dequeue}(Q)$

For all k where $k \in \text{fanin}(i)$ and $k \neq PI$

If ($T_l(k) = T_{max}$) and $V_t(k) = V_{th}$

Enqueue(Q, k)

selected = k

UpdateTimes(selected, V_{th}^L)

Until there exist $y \in \text{Critical path} \mid V_t(y) \neq V_{th}^L$

After initializing the graph with a high value for V_{th} to all the nodes, the $DualAssignment(V_{thi})$ algorithm selects a node which is on the critical path and then assigns a V_{th}^L to it. As this assignment may lead to a change in the critical path, it updates the critical path which is accomplished by the procedure $UpdateTimes(\text{selected}, V_{th}^L)$. The algorithm iterates till there exist at least one node on the critical path, which is yet to be assigned with V_{th}^L . The basic difference of the algorithm $DualAssignment(V_{thi})$ as proposed above with the algorithm $High-Vth-Assignment(V_{th2})$ as in [2] is summarized below:

The algorithm $High-Vth-Assignment()$ precedes in specific order (BFS-based backtracking from the primary outputs) of searching the nodes which can be assigned a higher threshold voltage. In fact, it distributes the *inherited* slack over a fewer number of gates than our algorithm $DualAssignment()$. Our algorithm $DualAssignment()$ searches the critical path (which keeps on changing) for a node with maximum saving and hence does not follow any particular order of searching.

Consider the graph of Fig. 2 as an example. The slack present with the node y is to be distributed among the nodes in its transitive fan-in cone. The algorithm $High-Vth-Assignment()$ as in [2] would assign the slack to node x itself while checking, if x can be assigned a high value for V_{th} . On the other hand, the $DualAssignment()$ assigns the slack not to x , but rather to each of i, j, k , and l . Thus the algorithm

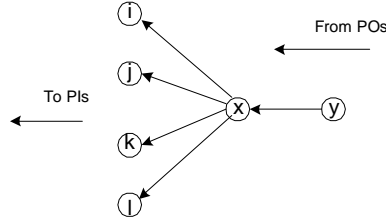


Fig. 2 Optimal assignment.

DualAssignment() assigns the higher threshold to **four** gates as opposed to **one** in [2].

3.3 Computation for Optimal High V_{th}

Due to the exponential relationship between threshold voltage and drain current in the weak inversion region, a higher threshold voltage will significantly reduce leakage current, thereby reducing leakage power. However, a higher threshold voltage will increase the equivalent on-resistance of each transistor, which results in a higher propagation delay. Normally, threshold voltage is empirically assumed to be $0.2V_{dd} \leq V_{th} \leq 0.5V_{dd}$. In dual threshold voltage applications, the values of V_{th}^L and V_{th}^H are very important. So far as the noise margin and other parameters are concerned, typical value of V_{th}^L is $0.2 V_{dd}$ [6]. Now, if we choose the value for V_{th}^H close to $0.2 V_{dd}$ then there is little difference of propagation delay between V_{th}^L and V_{th}^H transistors, and also it gives small leakage current improvement. On the other hand, if the value of V_{th}^H is close to $0.5 V_{dd}$, the leakage current can be reduced by a large amount for each such transistor. However, not many of gates on the off-critical path can be assigned the high threshold voltage. Thus, among the allowable values for high threshold voltages, there exist an optimal one. The next step is to calculate the optimal high threshold voltage ($opt_V_{th}^H$) corresponding to the best savings of standby leakage power. The algorithm *OptimalHighV_{th}H* is stated below, which, in fact, is very similar to that proposed in [2].

OptimalHighV_{th}H ()

```

i=1; Ri = Ro + ΔR
Calculate Vthi corresponding to Ri
While(Vthi < 0.5 Vdd)
  Initialization(Vthi)
  DualAssignment(Vthi)
  Estimate standby leakage power Pstdby
  If standby leakage power is the least power

```

$$P_{stdbymin} = P_{stdby}$$

$$opt_V_{th}^H = V_{thi}$$

$$++i; R_i = R_o + i.\Delta R$$

Calculate V_{thi} corresponding to R_i

Update network with $opt_V_{th}^H$

4.0 Implementation

The SIS (Berkeley) software tool has been employed to map the given circuit (in *blif* format) to a library of gates containing NAND, NOR and NOT gates. The mapping is performed with the option of minimum area for the given circuit. The first two types of gates have 2 to a maximum of 4 inputs. In the output file generated by the SIS tool, the circuit is described by equations employing the NAND, NOR and NOT operations. The mapped circuit is then translated into a bi-directed graph by using a parser. We have implemented our algorithm as well as the algorithm presented in [2] and compared results of the two algorithms for the standard ISCAS benchmark circuits. The value of various transistor parameters have been taken from the website <http://www.mosis.org> for the 74K series of transistors in $0.5 \mu m$ technology. The value of α is taken as 1.3 (for short channel MOSFETs).

5.0 Experimental Results

Experimental results as obtained are shown in Table-1. In Table-1, the first column represents the ISCAS Benchmarks used in our experiment. Second and third columns represent the standby leakage power with single $V_{th} = 2V_{dd}$ and the corresponding delay, respectively. Columns 4-8 represents the result based on the algorithm proposed in [2] and columns 9-13 give the result based on our algorithm.

Better results obtained by our algorithm is evident by comparing the highlighted columns in Table-1. Our algorithm assigns high threshold voltage value to more number of gates leading to about 86% average reduction in power compared to about 61% average reduction in power obtained in [2]. The power improvement is also better than that reported in [9].

6.0 Conclusion

In this paper we have proposed an algorithm for the assignment of high threshold voltage V_{th}^H to a maximum number of gates, and the value of V_{th}^H is selected such that there is maximum savings in power without compromise in performance. Because of the reason mentioned

Table 1

Bench mark	P _{stdby} Single V _{th} (μW)	Delay (ns)	Results based on [2]					Results based on our algorithm				
			Optimal V _{th} H (V)	Pstdby Dual V _{th} (μW)	# Trans. Assigned	% reduc tion	CPU time in Sec.	Pstdby Dual V _{th} (μW)	Optimal V _{th} H (V)	# Trans. Assigned	% reduc tion	CPU Time in Sec.
C432	19.78	7.98	0.337	6.23	546	69	20.9	1.87	0.337	892	91	32.4
C499	40.76	5.25	0.299	17.70	578	57	118.6	9.83	0.254	1512	76	174.1
C880	20.08	6.44	0.337	6.77	804	66	115.4	1.63	0.337	1278	92	165.6
C1355	56.71	6.11	0.337	27.84	542	51	198.8	15.28	0.254	1648	73	376.5
C1908	41.55	6.68	0.337	13.33	1028	68	225.0	7.50	0.299	1788	82	416.2
C2670	63.30	6.89	0.299	32.12	728	49	352.7	3.68	0.337	3094	94	609.8
C5315	138.15	8.59	0.337	45.26	4100	67	641.1	13.67	0.299	7016	90	1162.6
C7552	215.95	6.86	0.337	85.16	4028	61	964.2	22.91	0.299	9526	89	1853.2

in Sec 3.2, our algorithm assigns V_{th}^H to larger number of gates leading to more reduction in power than that of [2]. The time complexity of algorithm in [2] is $O(n)$. Actually, the order is that of BFS on a graph, and since the graph has maximum of $8n$ (maximum of 4 edges for forward and the same for backward) edges, the complexity of $O(|V| + |E|)$ degenerates to $O(|V|)$.

On the other hand, the time complexity of our algorithm is $O(n^2)$. This is because finding the critical path takes $O(n)$ time and in each iteration the critical path has to be found. As a consequence, our algorithm takes longer computation time. However, experimental results of Table I show that even for reasonably large benchmark circuits the overall time requirement is about double that of [2]. With the availability of increasingly faster computers, it does not appear to be a serious limitation.

References

1. R. X. Gu, and M. I. Elmasry, "Power Dissipation Analysis and Optimization of Deep Submicron CMOS Digital Circuits", IEEE JSSC, Vol. 31, No. 5. pp. 707-713, 1999.
2. L. Wei, Z. Chen, and K. Roy, "Design and Optimization of Dual Threshold Circuits for Low Voltage, Low Power Applications", IEEE Transaction on VLSI Systems, Vol. 17, No. 1, pp. 16-24, 1999.
3. L. Wei, Z. Chen, M. Johnson, K. Roy, and V. De, "Design and Optimization of Low Voltage High Performance Dual Threshold CMOS Circuits", Proc. 35th ACM/IEEE Design Automation Conference, pp. 589-492, 1998.
4. M. C. Johnson, K. Roy, and D. Somaskhar, "A Model for leakage control by transistor stacking", Technical Report TR-ECE.97-12, Purdue University, ECE Department.
5. J. Sheu, et al., "BSIM: Berkeley Short-channel IGFET Model for MOS transistors", IEEE JSSC, Vol.-22, No. 4, pp. 558-566, August 1987.
6. H. Oyamatsu, et al., "Design Methodology of Deep Submicron CMOS Devices for 1V Operation", IEICE Trans. Electron, Vol. E79-C, No. 12, pp. 1720-24, 1996.
7. N. H. E. Weste, and K. Eshraghian, "Principles of CMOS VLSI Design: A Systems Perspective, 2nd ed.", Addison Wesley, 1995.
8. S. Mutoh, et al., "1-V Power Supply High-Speed Digital Circuits Technology with Multithreshold-voltage CMOS", IEEE JSSC, Vol. 30, pp. 847-854, 1995.
9. Vijay Sundararajan, and Keshab K. Parhi, "Low Power Synthesis of Dual Threshold Voltage CMOS VLSI Circuits", Proc. ISLPED, pp. 363-368, 1999.