Level-Shifter Free Design of Low Power Dual Supply Voltage CMOS Circuits Using Dual Threshold Voltages

Abdulkadir U. Diril, Yuval S. Dhillon, and Abhijit Chatterjee
Georgia Institute of Technology, Atlanta, GA
{utku,yuvrajsd,chat}@ece.gatech.edu

Adit D. Singh
Auburn University, Auburn, AL
adsingh@eng.auburn.edu

Abstract

Usage of dual supply voltages in a digital circuit is an effective way of reducing power consumption due to the quadratic relation of supply voltage to power consumption. But the need for level shifters when a low voltage gate drives a high voltage gate has been a limiting factor preventing widespread usage of dual supply voltages in digital circuit design. The overhead of level shifters forces designers to increase the granularity of dual voltage assignment, reducing the maximum obtainable savings. We propose a method of applying dual supply voltages at gate level granularity without using level shifters. We modify the threshold voltage of the high voltage gates that are driven by low voltage gates in order to obtain the level shifting operation together with the logic operation. Using our method, we obtained an average of 20% energy savings for ISCAS'85 benchmark circuits.

1. Introduction

Energy consumption in CMOS circuits is proportional to the square of the supply voltage. This makes dual supply voltage usage popular for energy reduction. Since the speed of a gate decreases with decreasing supply voltage, dual supply voltage techniques [1][2][3][5][6][7][8] put low-voltage gates on the non-critical paths and high-voltage gates on the critical paths. This reduces the energy consumption in the low voltage gates while keeping the circuit delay unchanged.

Gate level dual supply voltage usage in CMOS circuits may suffer from excessive leakage energy if low voltage gates directly drive high voltage gates. In these situations, the PMOS transistor in the high voltage gate is not turned off completely with the low voltage “logic high” input signal. This leads to the use of level shifters wherever low voltage gates drive high voltage gates. To reduce or eliminate the delay, area, and energy overhead of the level shifters, researchers have proposed Clustered Voltage Scaling (CVS) [5][6] and Module Level Voltage Scaling (MLVS) [7][8]. In CVS, low voltage clusters are constructed in the circuit in such a way that there is no low voltage gate driving a high voltage gate. This is done by assigning low supply voltage to the gates starting from the circuit outputs depending on their slacks. Even though this method eliminates the use of level shifters, it cannot exploit the maximum energy savings possible with dual supply voltages. MLVS assigns the dual supply voltages to partitions of the circuit. This reduces the number of level shifters needed. Clearly this method also can not exploit the maximum energy savings possible with dual supply voltage usage since slacks of the gates in high voltage modules are not exploited.

There has also been research in the usage of gate level dual supply voltages with level shifters being used whenever a low voltage gate drives a high voltage gate [1][2][3]. In [1] and [2], graph theoretic algorithms are employed to apply dual supply voltage at the gate level while keeping the delay constant. [3] is an extension to CVS where level shifters are not restricted to be only in sequential elements. These algorithms try to minimize the number of level shifters used in the circuit to reduce their area, energy and delay overhead. However, apart from the area, energy and delay overhead, level shifters also decrease the slacks of the other gates in the same path, thereby reducing the number of gates that can operate with a lower supply voltage.

In this paper, we propose a circuit technique to eliminate the need for additional level shifter use in dual supply CMOS circuit design. We use a second threshold voltage in the PMOS transistors of the high voltage gates driven by low voltage gates, thereby providing them with built-in level-shifting capability. These modified gates have no energy or area penalties and only a slight delay penalty over the regular high voltage gates. The delay overhead is much less than a conventional level shifter delay, which is usually close to two inverter delays [4]. A similar approach for level shifter implementation is used in [4] where logic functionality is embedded into level
shifters. These level shifters have additional circuitry compared to a regular gate which results in area, delay, and energy overhead, whereas in our method, no additional circuitry is required.

The paper is organized as follows: Section 2 describes the dual threshold gate design for built-in level shifting. Section 3 gives the algorithm to determine which gates to apply the low supply voltage to. Section 4 explains the circuit modeling and simulation methodology. Section 5 gives the results of application of our technique on ISCAS’85 benchmark circuits. Section 6 concludes the paper.

2. CMOS Gate Design with Built-in Level Shifting Capability

The main need for level shifters in dual supply voltage CMOS circuits is to reduce the static leakage current in the high voltage gates when they are driven by low voltage gates. The low voltage output applied to the gate of the PMOS transistor in a high voltage gate is not enough to turn the PMOS transistor completely off. The slightly on PMOS transistor causes static current to flow from the power supply to the ground wasting significant energy. Level shifters are able to shift the voltage from a lower level to a higher one but since they do not perform any logic function, they cause area, delay and energy overhead.

To eliminate the overhead of additional level shifters, we use a second threshold voltage for the PMOS transistors in the high voltage gates which are driven by low voltage gates. By increasing the magnitude of the threshold voltage of the PMOS transistor, we decrease the static current flowing through the transistor when the gate voltage is $V_{DDL}$ (lower supply voltage). This increases the rise time for that gate slightly but depending on the second threshold voltage value (the magnitude of the original PMOS threshold voltage will be referred to as $V_{thp1}$ and the magnitude of the second PMOS threshold voltage will be referred to as $V_{thp2}$ from now on, $V_{thp2} > V_{thp1}$), we can decrease the static leakage current substantially.

Figure 1 shows the schematic for the proposed NAND2 gate with built-in level shifting capability. The darker line at the gate of M1 depicts the higher threshold voltage magnitude used for it. Such gates will be referred to as “Level Shifters” in the paper even though they are different from conventional level shifters. Figure 2 shows SPICE simulation results for static power dissipation of a level shifting NOT gate for different values of $V_{thp2}$ when the input voltage is at 1.4 Volts and the power supply voltage is at 1.8 Volts. 0.18µ SPICE Level 49 MOSFET models [9] were used for the simulations. The static power dissipations for a high voltage NOT gate (driven by another low voltage gate) are also given for comparison. It is seen that when $V_{thp2}$ is the same as $V_{thp1}$ (=0.4V), there is a lot of static power dissipation in the level shifting gate due to the inverter PMOS transistor not being off. Figure 3 and Figure 4 compare the propagation delay (average of delays for rising and falling output) and switching energy (the energy spent for 0 to 1 transition at the output) respectively for a high voltage, low voltage, and level shifting inverter for different values of $V_{thp2}$. The high supply voltage is again 1.8 Volts and the low supply voltage 1.4 Volts. The threshold voltage magnitude for a regular PMOS ($V_{thp1}$) is 0.4 Volts. Given that conventional level shifters have a delay close to two serially connected inverter delays [4], Figure 3 shows that a level shifting NOT gate will be much faster than a conventional level shifter followed by a high voltage NOT gate even when a high $V_{thp2}$ is chosen. This trend is similar for NAND and NOR gates as well.

Figure 1. Level shifting NAND2 gate with one high voltage and one low voltage inputs. M1 has higher threshold voltage magnitude than M2.

Figure 2. Static power dissipation of high voltage, low voltage, and level shifting NOT gates for different $V_{thp2}$ values.
3. Algorithm for Replacing High Supply Gates with Low Supply Gates

To reduce the energy consumption of combinational CMOS logic circuits, we propose replacing the high supply voltage ($V_{DDH}$) gates on the non-critical paths with slower, lower energy low supply voltage ($V_{DDL}$) gates. The total delay of the circuit remains the same as the original circuit which has only $V_{DDH}$ gates.

Algorithm $V_{DDH}$-$V_{DDL}$

Inputs: Topologically sorted list of circuit vertices, $V$;  
Output: Circuit with off-critical path $V_{DDH}$ gates replaced with $V_{DDL}$ gates.

For as many times as number of vertices {
    Let $v$ be the vertex with maximum metric value.
    high_delay←1, low_energy←0
    For every predecessor $p$ of $v$ {
        Depending on whether $p$ is mapped to $V_{DDH}$ or $V_{DDL}$, look-up “pin_delay” as the delay from this I/P of $v$ to the output using SPICE look-up table.
        Similarly compute pin_energy.
        high_delay←MAX(high_delay,pin_delay)
        low_energy←low_energy+pin_energy
    }  
    low_energy←low_energy/No. of predecessors $p$ of $v$
    If((high_delay-$v$.delay)$\leq$v.ts) {
        flag←0
        For every successor $p$ of $v$ {
            Compute largest pin delay of $p$ similar to above with $v$ mapped to a $V_{DDL}$ gate. Let this be “$p$.delay”.  
            If(($(v.es+high_delay)$>$p.es)) {
                If($(v.es+high_delay+p_delay)$$>$$p.lf)$
                    flag←1,break.
                }  
                Else if((($p$.delay-$p$.delay)$>$$p.ts)) flag←1,break.
            }
        }
        If(flag=0) {
            Map $v$ to a $V_{DDL}$ gate.
            $v$.delay←high_delay
            $v$.energy←low_energy
            $v$.metric←-2
            For every successor $p$ of $v$ {
                Compute new largest pin delay of $p$ and assign to $p$.delay
                Compute average pin energy of $p$ and assign to $p$.energy
            }
            Update_Time_Slacks($V$)
        }  
        Else $v$.metric←-1
    }  
    Else $v$.metric←-1
}

We first represent the combinational circuit as a directed acyclic graph (DAG), $G(V,E)$. If the circuit has multiple primary inputs (PIs), we create a dummy PI vertex, $P_{IL}$, which fans-out to the original PIs. Underlying this is the assumption that all inputs arrive simultaneously. Similarly for POs, we create a dummy PO vertex, $P_{OR}$, which has fan-ins from the original POs. Each vertex ‘$v$’ of the DAG has associated with it the following information:

- **Propagation delay** of high voltage, low voltage, and level shifting NOT gates for different $V_{thp2}$ values.
- **Switching energy** of high voltage, low voltage, and level shifting NOT gates for different $V_{thp2}$ values.

Figure 3.

Figure 4.

Figure 5.
1. The logic function computed by the gate corresponding to the vertex.
2. The supply voltage type of the gate (V_{DDH} or V_{DDL}).
3. The current delay (v.delay), energy (v.energy), time slack (v.ts), early start time (v.es), early finish time (v.ef), late start time (v.ls) and late finish time (v.lf) of the vertex at any stage of the replacement process. The delay/energy consumption values for V_{DDH} and V_{DDL} gates of different fan-ins/fan-outs/types-switching activities when driven by V_{DDH} or V_{DDL} gates are obtained from SPICE simulations as explained in Section 4. These values for a gate are updated whenever it or its driver gates change from type V_{DDH} to V_{DDL}. The dummy vertices have zero delays/energy consumptions. Please refer to [10] for definitions of early start time, etc.

The algorithm for replacing V_{DDH} gates with V_{DDL} gates consists of two steps:

1. Initialization: In this step, we first topologically sort the DAG to get the sorted vertex list, V^s. V^s is used to compute the total circuit delay, T, of the baseline circuit in which each vertex is mapped to a V_{DDH} gate, and to efficiently compute the time slack, early start time, early finish time, late start time and late finish time of each vertex using the function Update_Time_Slacks(V^s). In this step, we also compute the initial total energy of the baseline circuit with all V_{DDH} gates.

2. V_{DDH} to V_{DDL} replacement: First, with each vertex mapped to a V_{DDH} gate, we compute an energy metric for each vertex, which is just the energy saving obtainable if the vertex gate is changed from V_{DDH} to V_{DDL}. This metric is -1 for a vertex if either (i) the delay increase of the vertex due to the change is greater than the vertex’s time slack or (ii) the delay increase of any of the driven gates (due to the reduced drive provided by the V_{DDL} gate) is greater than its time slack. In case all the delay increases are less than the corresponding time slacks, the change in energy of the driven gates is also included in the metric for the driver gate. The metric value might still be negative (if energy is increased due to change from V_{DDH} to V_{DDL}), but it will be greater than -1 and hence the vertex will have higher priority for changing from V_{DDH} to V_{DDL} over vertices which have metric -1.

Next, we visit every vertex in decreasing order of the energy metric, and attempt to replace the gate corresponding to the vertex with the V_{DDL} equivalent. This might not always be possible, even for a vertex with non-negative metric value, because the metric for the vertex was computed under the assumption that all other vertices are mapped to V_{DDH} gates and hence the vertex had a lot of slack. As the replacement proceeds, the slack available to a vertex keeps on reducing and might not be sufficiently large to allow V_{DDH} \rightarrow V_{DDL} replacement. After every replacement, the slacks for all gates are recomputed using the function Update_Time_Slacks(V^s). The whole procedure is repeated till all vertices have been visited. Figure 5 gives the details of the algorithm used in this step.

After step 2 has been carried out, some V_{DDH} gates on off-critical paths have been replaced by V_{DDL} gates. This step does not change the total circuit delay since only those V_{DDH} gates which have sufficient time-slaack are replaced by V_{DDL} gates. The input PMOS transistors of the V_{DDH} gates that are driven by V_{DDL} gates are then modified to use the higher threshold voltage, V_{thp2} to reduce their static energy dissipation.

4. Methodology

We tested our scheme on the ISCAS’85 benchmark circuits. The circuits were synthesized using Synopsys Design Compiler to a target library that was reduced to have only two to four input ‘NAND’ and ‘NOR’ gates, and ‘INVERTER’ gates for simplicity. All the circuits were optimized for minimum delay.

To perform the energy optimization described in Section 3, we need the delay and energy consumption characteristics of the gates in the circuit. Lookup tables for dynamic and static energy and delay for different values of fan-out capacitances were generated for the gates used in the synthesis using 0.18µm SPICE Level 49 MOSFET models [9]. The standard supply voltage for this process (1.8 Volts) was used as V_{DDH}. High supply voltage (1.8V) gates were simulated by applying high voltage at their inputs. Low supply voltage gates and level shifters were simulated for a combination of high voltage and low voltage signals at their inputs. For example, for a NAND2 level shifting gate, there are 2 possibilities:

1. Both inputs are driven by low voltage. In this case, both 2 PMOS transistors have high threshold voltages (V_{thp2}).
2. One input is driven by high; the other input is driven by low voltage. In this case one PMOS transistor has V_{thp2} and the PMOS transistor which is driven by high voltage has V_{thp1}.

We applied inputs with switching activity 0.1 and static probability of 0.5 (of input being high) to all the primary inputs and used Synopsys Design Compiler to get the static probabilities and switching activities of the internal nodes. The average total energy consumption (i.e. static energy + dynamic energy) per clock cycle for gate ‘i’ was calculated as follows:

\[
\bar{E}_i = T_{slack} \left( prob^i \cdot P^d + \text{prob}^i \cdot P^s \right) + \text{Activity} \cdot \left( \frac{E^{st}_{thp1} + E^{st}_{thp2}}{2} \right) \quad (1)
\]
Where $T_{\text{clock}}$ is the clock period, $\text{prob}_i^0$ and $\text{prob}_i^1$ are the static probabilities of the output of gate ‘i’ being 0 and 1 respectively; $P_i^0$ and $P_i^1$ are the static power dissipations of gate “i” when the output of the gate is 0 and 1 respectively; and $E_{i}^{01}$ and $E_{i}^{10}$ are the energy dissipations of gate “i” when the output of the gate is switching from 0 to 1 and from 1 to 0 respectively.

5. Results

We implemented the algorithm in Figure 5 using C++. The compiled program was run for each of the benchmark circuits on a Sun Sparc Ultra-80 machine. Table 1 shows the optimization results for ISCAS’85 benchmark circuits. Initial energy is the average energy per clock cycle for the baseline circuit which has 1.8 Volt supply voltage for all gates. Final energy is the average energy per clock cycle for the optimized circuit which has some fraction of $V_{DDL}$ gates. The fraction of $V_{DDH}$ gates driven by $V_{DDL}$ gates are also given in Table 1. For small circuits, best energy reduction is obtained when $V_{thp2}$ is 0.4 Volts (i.e. level shifters are identical to high voltage gates). This is due to the fact that static energy dissipation is negligible in small circuits compared to the dynamic energy consumption. Even the high static energy dissipated in the high voltage gates driven by low voltage gates does not increase the total energy dissipation of the circuit substantially. As the circuit size increases (#gates > 1000), the optimum $V_{thp2}$ value increases to 0.5. We expect to see the $V_{thp2}$ increase more for even larger circuits.

We ran the algorithm explained in Section 3 for a range of $V_{DDL}$ and $V_{thp2}$ values. Figure 6 shows the variation of average energy savings for the ISCAS’85 benchmark circuits with $V_{DDL}$ varied from 1.1 to 1.6 Volts and $V_{thp2}$ varied from 0.4 to 0.9 Volts. Saving of “0” means that the algorithm couldn’t find any $V_{DDH}$ gates to swap with a $V_{DDL}$ equivalent. The best average energy saving of 19.81% was obtained for $V_{DDL}$ of 1.4 Volts and $V_{thp2}$ of 0.5 Volts.

The optimum $V_{DDL}$, $V_{thp2}$ and the low voltage assignment depends on the circuit size, structure and input switching activity. It is desirable to reduce the energy dissipation for a given circuit for a range of input switching activities. We simulated the circuits which were optimized for an input switching activity of 0.1 with different input switching activities. Figure 7 shows the average energy savings obtained for the benchmark circuits, which are optimized with input switching activity of 0.1, when different switching activities are applied to them. As seen in the figure, the circuits optimized for switching activity of 0.1 gives similar savings for different input switching activities as well. Even though the input switching activity is an important factor in the optimization, a circuit which is optimized for a particular switching activity still saves energy for other switching activities.

The presented method can be used together with dual threshold voltage assignment for static energy reduction. If the static energy optimization is applied before the dynamic energy optimization, the static energy optimizer should not exploit all the available slack in order to balance the static energy reduction and dynamic energy reduction. Similarly, if a very aggressive dynamic energy optimization is performed first, the static energy optimizer won’t be able to find any slack to exploit.
We presented a method to save dynamic energy in CMOS circuits using dual supply voltages without the need for additional level shifter gates. We use a second threshold voltage value for the PMOS transistors in the high voltage gates driven by the low voltage gates to reduce the static energy dissipation in those gates. These modified gates act like level shifters. This method gives an average energy saving of 19.81% for the ISCAS'85 benchmark circuits for switching activity of 0.1 at the circuit inputs. We show that the circuits which are optimized for input switching activity of 0.1 reduces dynamic energy for different input switching activities as well. Our method allows gate level dual supply voltage use without the area, delay, and energy overhead of level shifters.

REFERENCES


Table 1. Results for I/P switching activity of 0.1.

<table>
<thead>
<tr>
<th>Circuit</th>
<th># Gates</th>
<th>Delay (nsec)</th>
<th>Optimum VDDL (V)</th>
<th>Optimum Vthp (V)</th>
<th>Initial Energy (pJ/cycle)</th>
<th>Final Energy (pJ/cycle)</th>
<th>Fraction of Low VDD Gates</th>
<th>Fraction of High Vthp Gates</th>
<th>% Saving</th>
</tr>
</thead>
<tbody>
<tr>
<td>C432</td>
<td>332</td>
<td>1.22</td>
<td>1.5</td>
<td>0.4</td>
<td>0.49</td>
<td>0.42</td>
<td>0.48</td>
<td>0.19</td>
<td>14.56</td>
</tr>
<tr>
<td>C499</td>
<td>772</td>
<td>1.17</td>
<td>1.5</td>
<td>0.4</td>
<td>1.40</td>
<td>1.21</td>
<td>0.34</td>
<td>0.17</td>
<td>13.91</td>
</tr>
<tr>
<td>C1908</td>
<td>899</td>
<td>1.60</td>
<td>1.5</td>
<td>0.4</td>
<td>1.58</td>
<td>1.30</td>
<td>0.51</td>
<td>0.18</td>
<td>17.88</td>
</tr>
<tr>
<td>C2670</td>
<td>934</td>
<td>0.92</td>
<td>1.4</td>
<td>0.4</td>
<td>1.52</td>
<td>1.18</td>
<td>0.59</td>
<td>0.15</td>
<td>22.29</td>
</tr>
<tr>
<td>C3540</td>
<td>1395</td>
<td>1.79</td>
<td>1.4</td>
<td>0.5</td>
<td>2.28</td>
<td>1.82</td>
<td>0.59</td>
<td>0.14</td>
<td>20.41</td>
</tr>
<tr>
<td>C5315</td>
<td>2106</td>
<td>1.50</td>
<td>1.3</td>
<td>0.5</td>
<td>3.64</td>
<td>2.61</td>
<td>0.64</td>
<td>0.12</td>
<td>28.21</td>
</tr>
<tr>
<td>C7552</td>
<td>2846</td>
<td>1.30</td>
<td>1.4</td>
<td>0.5</td>
<td>5.48</td>
<td>4.05</td>
<td>0.62</td>
<td>0.13</td>
<td>26.06</td>
</tr>
</tbody>
</table>