HA²TSD: Hierarchical Time Slack Distribution for Ultra-Low Power CMOS VLSI

Kyu-won Choi and Abhijit Chatterjee
School of Electrical and Computer Engineering
Georgia Institute of Technology, Atlanta, GA 30332
{kwchoi,chat}@ece.gatech.edu

ABSTRACT
This paper describes an efficient hierarchical design and optimization approach for ultra-low power CMOS logic circuits. We introduce the *Hierarchical Activity-Aware Time Slack Distribution* (**HA²TSD**) algorithm, which distributes the surplus time slack into the most power-hungry modules hierarchically. **HA²TSD** ensures that the total slack budget is maximal and the total power is near-minimal. Based on these time slacks, we have optimized technology parameters (supply voltage, threshold voltage, and device width) through a gate-level power optimizer and have tested the algorithm on a set of benchmark example circuits and building blocks of a synthesizable ARM core. The experimental results show that our strategy delivers over an order of magnitude savings in total (static and dynamic) power and reduces the optimization run-time significantly.

Categories and Subject Descriptors
B.7.2 [Integrated Circuits]: Design Aids-simulation.
General Terms
Algorithms.
Keywords
Low-power design, time slack distribution, and gate-level power optimization.

1. INTRODUCTION
Recent advances in wireless networking technology and the rapid development of semiconductor technology have introduced new challenges in the design of portable devices such as personal digital assistants (PDAs). Power optimization for those embedded systems and power constrained mobile computing is an active area of research that has received considerable attention in most recent years. Delay, area and power trade-offs for complex systems require the use of advanced algorithms and EDA tools. To achieve excellent power and performance results, future EDA tools must harness the combination of technology parameters, i.e., multiple supply voltages (Vdd), multiple threshold voltages (Vth), and transistor resizing (W). By combining the optimization strategy with the on-the-fly technology parameter scaling, designers and EDA tools can fully explore the design space of dynamic power, static power, and timing slack [1,2].

In general, low-power optimizations that do not compromise performance are dependent on time slack calculation and the surplus delay (slack budget) distribution among the circuit modules. Time slack is measured as the difference between the signal required time and the signal arrival time at the primary output of each module. The first use of the slack distribution approach was reported by the popular *zero-slack algorithm* (**ZSA**) [3]. The **ZSA** is a greedy algorithm that assigns slack budgets to nets on long circuit paths. It ensures that the net slack budget is maximal, which means that no more slack budget can be assigned to any of the nets without violating the path timing constraints. Most other slack distribution algorithms are pruning versions of **ZSA** [4,5] for improving *delay performance of circuits*. However, the objective of the timing analysis in this paper is to provide a low-power methodology that maintains the high speed of circuits. The **HA²TSD** algorithm is different from the **ZSA** in three principal aspects: i) time slack distribution of each module is based on power rather than performance metrics; ii) the slack distribution is performed hierarchically, and iii) the technology parameters of each module are optimized at the gate level.

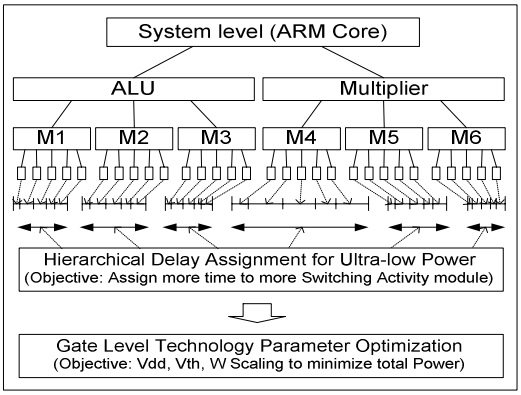


Figure 1. Hierarchical Delay Assignment and Gate Level Power Optimization

2. DELAY AND ENERGY MODEL
We use a transregional model for estimating the worst-case signal propagation delay through a gate. The delay model has been derived using an extension of the alpha-power law saturation drain current model [7] to the subthreshold region. The drain current model incorporates effects of high-field and quasi-ballistic (velocity overshoot) carrier transport in the MOSFET channel. The delay model consists of four major components: 1) the delay due to switching MOSFETs, 2) the distributed interconnect RC delay, 3) the time of flight delay, 4) the delay component due to the non-zero rise time of the input signal are considered. These definitions of gate delay and interconnect resistance delay allow the definition of arrival

ISLPED'02, August 12-14, 2002, Monterey, California, USA.

times and required times at the input and output of a gate in the network, which are used for defining time slack.

$$t_{d_v} = \left[\frac{1}{2} - \frac{1-\frac{V_{TS_v}}{V_{dd}}}{1+\alpha}\right] \max_{i\in(1,f_{ii}(v))}\{t_{d_{i,v}}\} + \frac{V_{dd}/2}{I_{Dvw} - f_{ii}(v)\beta I_{off}} \cdot \left[C_{DP_v} + \frac{1}{w_v}\sum_{j=1}^{f_{oi}(v)}(w_{vj}C_{t_{vj}} + C_{INT_{vj}})\right]$$

$$+ \max_{j\in(1,f_{oi}(v))}\{t_{d_{v,j}}\}\left[R_{INT_{vj}}(w_{vj}C_{t_{vj}} + \frac{1}{2}C_{INT_{vj}}) + \frac{L_{INT_{vj}}}{v_{INT}}\right] + \frac{1}{2}C_{m_v}V_{dd}\sum_{j=1}^{f_{ii}(v)-1}\frac{1}{I_{Dvw}(j)} \quad (1)$$

In the above equation, V_{dd} is the power supply voltage, t_{dv} is the delay of gate G_v, V_{TSi} is the threshold voltage of the ith gate, α is the velocity saturation coefficient ($1 \le \alpha \le 2$), $t_{di,v}$ is the delay of the gate G_v at the ith fan-in, $t_{dv,j}$ is the delay of the gate G_v at the jth fan-out, $I_{Dvw}(f_{ii})$ is the switching drain current per unit width, f_{ii} is the number of fanins, f_{oi} is the number of fanouts, β is the pMOSto nMOS width ratio ($\beta \ge 1$), I_{off} is the off current per unit width, C_{DPv} is the sum of the overlap, junction and finging capacitance at the output node per unit width, w_v is the device width, adjusting w_v scales the widths of all the transistors in G_v ($w_v \ge 1$), w_{vj} is the device width the gate at the jth fan-out ($w_{ij} \ge 1$), C_{tvj} is the input capacitance per unit width of the gate being driven by the jth fan-out, C_{INTvj} is the interconnect capacitance at the jth fan-out, R_{INTvj} is the interconnection resistance at the jth fan-out, L_{INTvj} is the interconnection length at the jth fan-out, v_{INT} is the propagation velocity through the interconnect, C_{mv} is the intermediate node capacitance of series connected MODFET's in multiple fan-in gates, f_c is the clock frequency, η_v activity factor of the gate output, and K_{SC} is the coefficient for short-circuit dissipation [8]. The models are described in detail in our previous work [6].

The equations used to compute the dynamic and static energy dissipations of a gate are described next. Similar models have been presented and analyzed in a recent work by [8]. It is assumed that the gates are simple multi-input gates with symmetric series or parallel pull-up and pull-down MOSFET configurations. Contributions of subthreshold leakage through the MOSFET channel as well as the leakage across the device drain junctions to static dissipation are included.

1) Static Dissipation of Gate G_v ($v \in N$):

$$E_{Static_v} = V_{dd} W_v I_{off} / f_c \quad (5)$$

2) Dynamic and Short-Circuit Dissipation of G_v

$$E_{Dynamic_v} = \frac{1}{2}\eta_v V_{dd}^2 (1+K_{SC}) \cdot \left[w_v\{C_{DP_v} + (f_{ii}(v)-1)C_{m_v}\} \sum_{j=1}^{f_{oi}(v)}(w_{vj}C_{t_{vj}} + C_{INT_{vj}})\right] \quad (6)$$

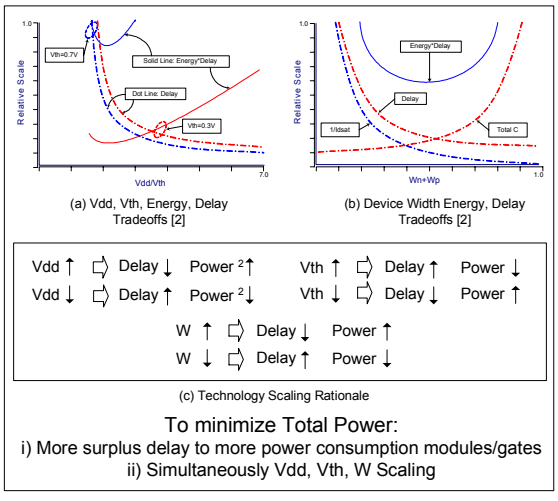
3. PREVIOUS WORK

Supply voltage scaling technique for low power has been investigated in almost all levels of the design hierarchy from system level to device level due to the quadratic effect on the switching power component. Many respective researches have been shown up in literature [1]. However, it does not come without penalties [9]. The scaling limitations of Vdd reduction are: 1) Delay increase (performance requirements impose a limit); and 2) Noise margins decrease (circuit is more susceptible to noise related soft failures). The approaches to overcome the extent of Vdd scaling are: 1) Availability of high-efficiency DC-DC converter for use [10]; 2) Scaling down the dimensions of devices along with Vdd to compensate for the effects of Vdd on performance; and 3) Reduction of the threshold voltage of transistors.

Threshold voltage scaling can be used to compensate the performance penalty of the Vdd reduction. In addition, for the active mode of operation, the low Vth is preferred because of the higher performance. However, for the standby mode, high Vth is useful for reduction of leakage power. Different threshold voltages can be developed by multiple Vth implantation during the fabrication, by changing the substrate and source bias, by controlling the back gate of double-gate SOI (silicon on insulator) devices [10]. Some techniques in literature are: 1) SATS (self adjusting threshold voltage scheme) [11]; 2) MTCMOS (multi-threshold voltage CMOS) [12]; 3) DTMOS (dynamic threshold voltage MOSFET) [13]; and 4) DGDT-SOI (double fate dynamic threshold control SOI) [14]. In general, the threshold voltage is a function of a number of parameters including the following: 1) Gate conductor, 2) Gate insulation material, 3) Gate insulator thickness-channel doping, 4) Impurities at the silicon-insulator interface, and 5) Voltage between the source and the substrate.

Transistor and gate sizing affects for dynamic and leakage power reduction and delay. A large gate is required to drive a large load capacitance with acceptable delay but requires more power. The basic rule is to use the smallest transistors or gates that satisfy the delay constraints. To reduce dynamic power, the gates that toggle with higher frequency should be made smaller. An interesting problem occurs when the sizing goal is to leakage power of a circuit. The leakage current of a transistor increases with decreasing threshold voltage and channel length. In general, a lower threshold or shorter channel transistor can provide more saturation current and thus offers a faster transistor. This presents a tradeoff between leakage power and delay. There have been a number of optimization algorithms for gate sizing for dozens of years [15].

Figure 2 presents the fundamental characteristics of those three device parameters (Vdd, Vth,W) for power and delay tradeoffs [2]. Figure 2(a) shows the Vdd/Vth and Delay*Energy tradeoffs. It shows that the supply voltage should be larger than four times of the threshold voltage if the delay is not to increase excessively. Figure 2(b) shows the Device Width and Delay*Energy tradeoffs. It is shown that the delay decreases with increase device width but the delay-energy product is minimized when the devices contribute half of the total load capacitance. The technology parameters trade-offs are summarized in Figure 2(c). In this paper, we try to optimize the non-linear parameters of those tradeoffs efficiently to minimize the total power.



(a) Vdd, Vth, Energy, Delay Tradeoffs [2]

(b) Device Width Energy, Delay Tradeoffs [2]

(c) Technology Scaling Rationale

Figure 2. Technology Optimization Rationale

4. PROPOSED APPROACH

The key steps of our approach are shown in Figure 3. First hierarchical circuit partitioning is performed. Then, beginning with the topmost level of the design hierarchy, delay values are assigned to every module at that level. The total delay from PI to PO is given. The problem is to determine the delays of the individual modules so that total power consumption can be minimized by optimizing the supply voltage, threshold voltage and device sizes of module M_j *for the assigned delay values*. The procedure is repeated hierarchically. We use the following heuristic to assign delays to each module.

Heuristic: In a given data flow graph of M_j modules, let $C_j = \sum_{node\ i} \eta_i c_i$ be the summation of the product of the activity η_i at node i and the capacitance c_i at node i over all nodes i of the module M_j. If the delay assigned to module M_j is D_j, then the best delay assignment for minimizing power is obtained when

$$\frac{D_1}{C_1} = \frac{D_2}{C_2} = \cdots = \frac{D_j}{C_j}$$

It is clear that such an assignment of delay to each M_j can cause the overall path delay constraint (sum of delays assigned to each module) to be violated for some of the paths in the module. Therefore, the iterative **HA^{2}TSD** algorithm is used to solve the problem. This is described below.

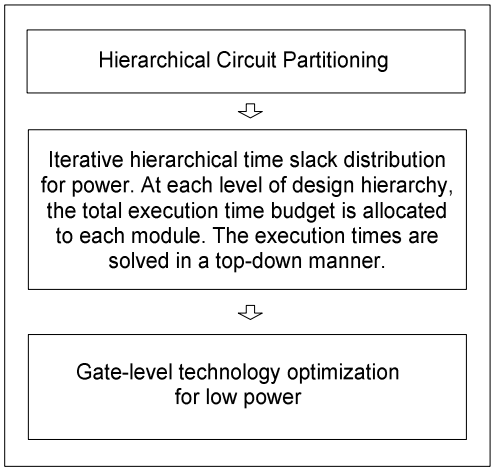


Figure 3. Power Optimization Procedure

4.1 Topological Depth-Based Partitioning

For simulation run-time efficiency and power optimization effectiveness, we introduce a circuit partitioning algorithm which ensures the minimization of the delay skew between sub-modules, and constrains maximum sub-module size (or fan-out size). Figure 4 gives conceptual overview of the topological depth-based partitioning. First of all, labeling of each circuit node is conducted according to the topological order. Then, according to the maximum depth and maximum size constraints, the whole flattened gate-level digital circuit is partitioned into sub-module circuits. The detailed algorithm for the partitioning is shown in Figure 5. The complexity of this algorithm is $O(b^m)$, where b is the branching factor (i.e., average fan-out number) and m is maximum topological depth.

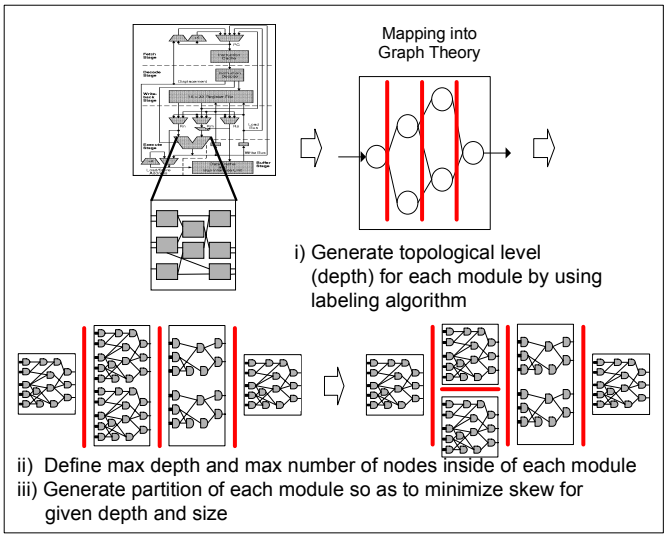


Figure 4. Partitioning Overview

HA2TSD Algorithm 1 : Partitioning

Input: *Directed acyclic graph* **G** = (**V,E**)
Output: *Partitioned sub-graphs* $G_i (V_i, E_i)$, *i= 2 … n*

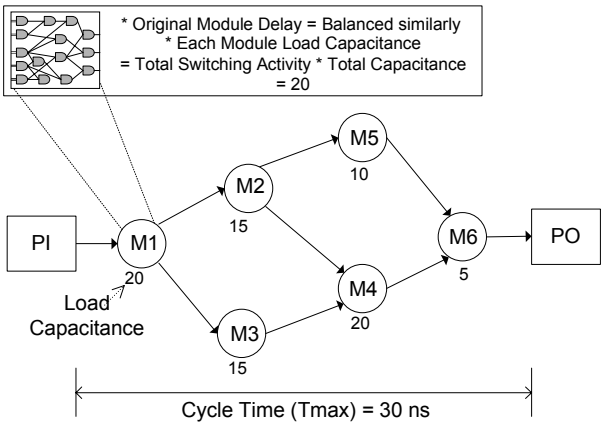
```
Begin
 Phase 0: Initialization
   Initialize the class variable of all nodes V in G ;
   (level (topological order) = ∞ )
 Phase I: Labeling {Identify topological order}
   Put a start node to FIFO Queue q;
   Initialize the level of the start node = 0;
   While (q Is not 0)
   {
       Obtain a reference to the first element(x) in q;
       Remove node x from q;
       For each fan-out node y of node x
       {
          If level of y = ∞ then
          {
             If number of fan-in node of y =1 then
                level of y = level of x +1;
             else if number of fan-in node of y >1 then
                level of y =MAX( levels of fan-in nodes of y) +1;
             Add node y into q;
          }
       }
   }
 Phase II: Partitioning
   Sort all nodes according to the topological order;
   Partition the graph G by constraints (max node number & depth);
End
```

Figure 5. Partitioning Algorithm

4.2 Activity-Aware Delay Assignment

Figure 6 presents an example of the module level delay assignment algorithm. In the first step, each module is sorted by the amount of load capacitance of each module (step 1). According to the priority of each module, we assign maximum delay with the "objective function" and "delay assignment" formula in Fig. 6 (Step 2 and 3). Then we look at the local improvement by local search (step 4). If all modules' delays are assigned, conduct the technology parameter optimization at the gate level (step 5). Finally, we generate the power/area saving values and optimal parameters. In the algorithm,

each module (***M1,…,Mi***) can be a functional module or a sub-partition, the total physical capacitance of a module can be the sum of the fan-in/out counts inside the module, and the load capacitance of each module can be calculated by multiplying the total switching activities by the total fan-in/out net counts. Its algorithm is shown in Figure 7. The complexity of the algorithm is $O(nb^m)$, where n is the number of modules, b is the branching factor (i.e., average fan-out number) and m is maximum topological depth.



Object : Assign max delay of each module for max power saving
(Note: slack time = power/area saving)

Object Function = $\frac{D_1}{C_1} = \frac{D_2}{C_2} = \cdots = \frac{D_6}{C_6}$

$D_1 + D_2 + \cdots + D_6 \le T_{\max}$

$C_j = \sum_{node\ i} \eta_i c_i$

(η_i = switching activity at node i, c_i = capacitance at node i)

Delay ssignment = $\frac{C_j}{\text{Total Load Capacitance Sum in Path}} \times (T_{\max}$ - Assigned Delay Sum)

Step 1: Module Priority queue for each module by load capacitance

M4, M1 (20)
M2 (15)
M3 (10)
M5,M6 (5)

Step 2: - Select M4
- Path Priority queue for each path with M4

Path1: M1,M2,M4,M6 (60)
Path2: M1,M3,M4,M6 (55)

- Select Path1
- Delay of M4 = (20/60)*30 = 10 ns

Step 3: - Repeat Step2 for all modules
- Delay of M1 = (20/40)*20 = 10 ns
- Delay of M2 = (15/20)*10 = 7.5 ns
- Delay of M3 = (10/15)*10 = 6.66 ns
- Delay of M5 = 10 ns
- Delay of M6 = 2.5 ns

Step 4: - Local search improvement
- Increase 6.66 to 7.5 for M3

Step 5: - Go to Gate level optimization (Vdd, Vth, W Scaling) with this Max delay of each module

Figure 6. An Example of Delay Assignment

HA2TSD Algorithm 2 : Delay Assignment

Input: *Partitioned sub-graphs* $G_i(V_i, E_i)$
Output: *Delay weighted sub-graphs* $G_i(V_i, E_i, W(v_i))$

Begin
Phase 0: *Initialization*
Enumerate the critical paths P_j *in* $G = \{G_1 \ldots G_i\}$;
Sort P_j *in decreasing order of criticality;*
Phase I: *Delay assignment for each path*
Identify maximum delay T_{max} *of all paths;*
Calculate switching activity α_i *for all nodes* V_i
Set the delay for nodes V_i *on critical path(s)*
While *(all path* P_j *)*
{
While *(unassigned* $V_i = 0$ *)*
{
$T_{max}(V_i) = [\alpha_i / (\alpha_1 + ..+\alpha_{i-1} + \alpha_{i+1}+..+ \alpha_n)] * T_{max}$;
/* where n = number of nodes on the P_j */
$W(V_i) = T_{max}(V_i)$;
}
}
End

Figure 7. Delay Assignment Algorithm

4.3 Gate-level Power Optimization

There are three ways to save power dissipation while maintaining operation frequency by utilizing surplus time slack in non-critical paths: i) employing multiple-Vdd to lower supply voltage, ii) employing multiple-Vth to reduce leakage current, and iii) employing multiple-W to reduce circuit capacitance. In this paper, *the Vdd reduction is main scaling parameter for low power, and Vth and W scaling is mainly for creating more time slack for the ultra-low power optimization*. The difficulties of the power optimization at gate level come from two major aspects: i) the non-linear interactions of the object parameters, for example, each gate has at least four non-linear variables (Vdd, Vth, W, Delay) and ii) the optimization time complexity, for example, after logic synthesis of target system, each functional module (i.e., ALU, Adder, Multiplier, etc.) might generate large number of gates/interconnections and the searching space for the optimization is exponential. Therefore, simulation-efficient partitioning scheme should be judiciously considered before the gate level optimization. The Figure 8 shows the relationship between the maximum delay assignment and the technology scaling for power savings.

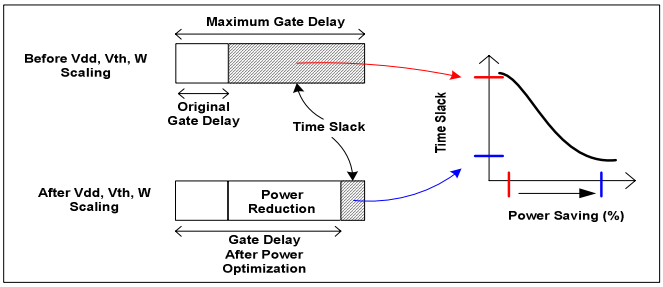


Figure 8. Time Slack and Power Saving

After the maximum delays have been assigned to each module/gate in the circuit, we optimize each gate individually for minimum power. The strategy is to find iteratively, using binary search, the optimal combination of Vdd, Vth, and W for each gate that meets the

maximum delay condition while achieving minimum power dissipation. We used our previous work for the gate level power optimization [6]. This strategy is based on the observation that power consumption and delay are monotonic functions of Vdd, Vth, and W, individually, other parameters being fixed. Since it is impractical to have more than one power supply or threshold voltage in the circuit, we keep only one global value of Vdd and Vth. However, the algorithm could be easily modified to allow the use of multiple threshold values in the circuit if desired. The algorithmic complexity of this procedure depends on the number of iteration steps that we allow for convergence to the optimal values. Assuming that V_{DD}, V_{th} and W are each constrained to 2^M quantized values, it takes $O(M^3)$ simulations of the entire circuit to obtain the final optimal values. This is many orders of magnitude lower than the complexity of any direct or random search algorithm that may be used to search for the optimal solution.

5. RESULTS

We developed a simulation frame work with C/C++/STL and Perl on Ultra-80 Unix machine for the hierarchical power optimization. Also, we used off-the-shelf commercial tools for the RTL description, the functional verification, and the logic synthesis of the target system. A few arithmetic modules from the target system and ISCAS89/MCNC91 benchmark circuits are used for the experimental demonstration. For the range of the technology parameter values, the 2001 updated version of ITRS (International Technology Roadmap for Semiconductors) and the MOSIS (Integrated Circuit Fabrication service) parameter test results with TSMC 0.25 micron are used. For the RTL design, we used verilog hardware description, for the functional simulation, we used VCS (synopsys), and for the logic synthesis, we used design analyzer (synopsys) with 0.25 micron TSMC library.

Monte Carlo simulation is performed for activity profiling of each module/sub-module as described in [2]. This approach consists of applying randomly generated input patterns at the primary inputs of the circuit and monitoring the switching activity per time interval T using a simulator. Under the assumption that the switching activity of a circuit module over any period T has a normal distribution, and for a desired percentage error in the activity estimate and a given confidence level, the number of required simulation vectors is estimated. The simulation based approach is accurate and capable of handling various device models, different circuit design styles, single and multi-phase clocking methodologies, tristate drives, etc.

Figure 9 shows the hierarchy and the granularity that we used in our simulation. In this paper, we only simulated 3-level hierarchical case. Table 1(a) shows the total power consumption with fixed technology parameters for the given circuits. Table 1(b) demonstrates the efficiency and effectiveness of the hierarchical power optimization with the proposed design flow. The experimental results show that our power optimization strategy delivers an order of magnitude savings in total (static and dynamic) power without performance degradation over non-optimized benchmark circuits and our hierarchical approach is much faster than traditional approach. With the hierarchical depth of 3 as shown in Figure 9, we can obtain average 6 times faster optimization than the totally flattened case when we still have average 83.6% power savings.

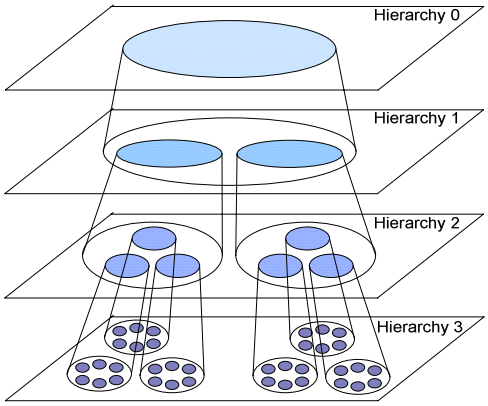


Figure 9. Hierarchy in our Simulation

6. CONCLUSION

This paper presents an efficient hierarchical low-power design flow and a novel switching activity based optimization algorithm for ultra-low power CMOS VLSI. Experimental results show that the algorithm yields reductions in power by typically a factor from 19.6x to 52.4x with optimal Vdd/Vth and multiple W scaling. In summary, key contributions of the new power minimization technique is: i) without compromising the speed, the total (static and dynamic) power is minimized significantly; ii) with the hierarchical approach, polynomial time optimization is feasible in very large circuits; and iii) the activity-aware delay assignment ensures that the total time slack is maximum and the total power is near-minimal. Future work will include application-specific and architecture-driven issues with this technology scaling techniques.

Table 1. Results of H^{2}TSD-Based Power Optimization

(a) Before Optimization (Fixed Vdd:3.3v, Vth:0.7v)

System Module	Gates/ Depth	Delay (ns)	Input Activity	ϖ, σ_ω	Power Dissipation			
					Leakage	Switching	Short-ckt	Total
4 - Full Adder	106/48	3.36	0.5	17.9, 24.5	2.09x10E-20	4.37x10E-11	2.15x10E-12	4.59x10E-11
			0.05	17.9, 24.5	2.09x10E-20	4.33x10E-12	2.13x10E-13	4.54x10E-12
16 - Look ahead	1838/ 81	7.0	0.5	5.9, 6.2	1.48x10E-19	7.65x10E-10	9.33x10E-11	8.58x10E-10
			0.05	5.9, 6.2	1.48x10E-19	1.39x10E-10	9.29x10E-12	1.48x10E-10
64 - ALU	3417/ 226	18.6	0.5	6.1, 9.2	1.12x10E-18	4.4x10E-09	2.87x10E-10	4.69x10E-09
			0.05	6.1, 9.2	1.12x10E-18	1.90x10E-10	2.87x10E-12	1.93x10E-10
s298	286/18	3.02	0.5	4.8, 7.7	1.92x10E-20	1.44x10E-11	2.37x10E-13	1.46x10E-11
			0.05	4.8, 7.7	1.92x10E-20	1.39x10E-13	2.55x10E-15	1.42x10E-13
s344	229/28	3.86	0.5	15.9, 26.2	4.59x10E-20	6.38x10E-11	9.87x10E-13	6.48x10E-11
			0.05	15.9, 26.2	4.59x10E-20	6.39x10E-13	9.62x10E-15	6.49x10E-13
s386	426/23	3.99	0.5	10.9, 9.2	5.56x10E-20	4.88x10E-11	9.99x10E-13	4.98x10E-11
			0.05	10.9, 9.2	5.56x10E-20	5.13x10E-13	9.62x10E-15	5.23x10E-13
s526	596/18	4.3	0.5	5.2, 7.8	5.88x10E-20	5.13x10E-11	2.00x10E-12	5.33x10E-11
			0.05	5.1, 7.8	5.88x10E-20	5.32x10E-13	9.82x10E-15	5.41x10E-13
c6288	2406/ 129	10.6	0.5	4.7, 8.2	6.52x10E-18	3.21x10E-09	6.55x10E-10	3.87x10E-09
			0.05	4.3, 8.1	6.52x10E-18	4.39x10E-10	6.54x10E-12	4.76x10E-10

(b) After Optimization (Vdd:0.6-1.2v, Vth:0.1-0.52v)

System Module	Hierarchy	Granularity	Input Activity	Vdd,Vth	ϖ, σ_ω	Total Power	Savings	
							Power	Run-Time
4- Full Adder	0	Each gate	0.5	0.6, 0.1	6.61, 8.14	1.95x10E-11	57.5%	0x
			0.05	0.7, 0.1	5.62, 7.19	3.14x10E-13	31.0%	0x
	2	Level 1: 53 Level 2: 17.7	0.5	0.625, 0.1	6.62, 3.14	2.95x10E-11	35.7%	4.28x
			0.05	0.725, 0.2	6.99, 5.64	3.57x10E-13	21.5%	4.28x
	3	Level 1: 53 Level 2: 17.7 Level 3: 2.9	0.5	0.7, 0.12	7.3, 6.22	3.19x10E-11	30.4%	18.8x
			0.05	0.725, 0.2	8.6, 9.0	3.67x10E-13	19.4%	18.8x
16- Look ahead	0	Each gate	0.5	0.8, 0.1	3.1, 2.14	2.93x10E-11	96.6%	0x
			0.05	0.825, 0.1	3.66, 2.94	8.03x10E-13	90.7%	0x
	2	Level 1: 919 Level 2: 306.3	0.5	0.8, 0.1	4.19, 1.14	3.09x10E-11	96.4%	2.26x
			0.05	0.825, 0.1	4.45, 7.10	8.66x10E-13	90.0%	2.26x
	3	Level 1: 919 Level 2: 306.3 Level 3: 51.1	0.5	0.8, 0.1	5.01, 4.14	6.40x10E-11	92.5%	3.08x
			0.05	0.85, 0.12	4.91, 6.16	1.02x10E-12	88.2%	3.08x
64-ALU	0	Each gate	0.5	0.9, 0.1	5.71, 3.13	5.26x10E-11	98.9%	0x
			0.05	0.925, 0.1	5.91, 5.10	2.34x10E-12	98.8%	0x
	2	Level 1: 1708 Level 2: 569.5	0.5	0.925, 0.1	3.63, 3.13	5.50x10E-11	98.8%	2.10x
			0.05	0.95, 0.12	4.62, 5.12	9.60x10E-12	95.0%	2.10x
	3	Level 1: 1708 Level 2: 569.5 Level 3: 94.9	0.5	0.95, 0.12	3.51, 8.15	8.09x10E-11	98.3%	2.70x
			0.05	0.925, 0.2	5.81, 6.14	2.30x10E-11	88.1%	2.70x
s298	0	Each gate	0.5	0.6, 0.1	2.62, 4.44	2.52x10E-13	98.3%	0x
			0.05	0.625, 0.1	3.21, 7.14	4.46x10E-15	96.8%	0x
	2	Level 1: 143 Level 2: 47.7	0.5	0.625, 0.1	3.61, 3.14	5.51x10E-13	96.2%	3.13x
			0.05	0.625, 0.1	3.31, 4.19	1.09x10E-14	92.3%	3.13x
	3	Level 1: 143 Level 2: 47.7 Level 3: 7.94	0.5	0.625, 0.1	4.11, 4.14	8.55x10E-13	94.2%	6.48x
			0.05	0.65, 0.12	4.31, 2.94	1.45x10E-14	89.8%	6.48x
s344	0	Each gate	0.5	0.7, 0.1	8.61, 9.34	6.44x10E-13	99.0%	0x
			0.05	0.725, 0.2	9.21, 3.14	8.31x10E-14	87.2%	0x
	2	Level 1: 115 Level 2: 38.16	0.5	0.8, 0.12	12.1, 5.14	2.03x10E-12	96.9%	3.32x
			0.05	0.8, 0.12	9.61, 2.14	9.36x10E-14	85.6%	3.32x
	3	Level 1: 115 Level 2: 38.16 Level 3: 6.36	0.5	0.85, 0.1	7.61, 3.15	6.02x10E-12	90.7%	7.62x
			0.05	0.825, 0.2	9.61, 2.14	1.35x10E-13	79.2%	7.62x
s386	0	Each gate	0.5	0.6, 0.1	4.61, 5.14	4.43x10E-13	99.1%	0x
			0.05	0.6, 0.1	5.88, 3.74	1.82x10E-14	96.5%	0x
	2	Level 1: 213 Level 2: 71	0.5	0.6, 0.1	7.61, 9.10	4.63x10E-13	99.1%	2.86x
			0.05	0.625, 0.1	7.61, 4.14	1.92x10E-14	96.3%	2.86x
	3	Level 1: 213 Level 2: 71 Level 3: 11.8	0.5	0.625, 0.1	9.33, 4.14	9.58x10E-13	98.1%	5.13x
			0.05	0.65, 0.12	10.01, 9.1	2.16x10E-14	95.9%	5.13x
s526	0	Each gate	0.5	0.6, 0.1	3.61, 3.34	6.91x10E-13	98.7%	0x
			0.05	0.625, 0.1	4.55, 5.15	1.84x10E-14	96.6%	0x
	2	Level 1: 298 Level 2: 99.3	0.5	0.625, 0.1	4.21, 2.14	1.00x10E-12	98.1%	2.68x
			0.05	0.625, 0.1	4.21, 5.94	2.46x10E-14	95.4%	2.68x
	3	Level 1: 298 Level 2: 99.3 Level 3: 16.6	0.5	0.625, 0.1	5.61, 6.14	1.93x10E-12	97.4%	4.39x
			0.05	0.65, 0.12	4.91, 7.14	3.62x10E-14	93.3%	4.39x
c6288	0	Each gate	0.5	0.925, 0.1	5.61, 3.14	3.49x10E-11	99.1%	0x
			0.05	0.9, 0.12	4.67, 4.44	7.10x10E-12	98.5%	0x
	2	Level 1: 1203 Level 2: 401	0.5	0.925, 0.1	3.91, 5.14	5.56x10E-11	98.6%	2.19x
			0.05	0.825, 0.1	5.69, 4.14	7.97x10E-12	98.3%	2.19x
	3	Level 1: 1203 Level 2: 401 Level 3: 66.8	0.5	0.95, 0.2	4.61, 6.99	8.81x10E-11	97.9%	2.90x
			0.05	0.925, 0.2	5.71, 5.54	7.03x10E-11	85.2%	2.90x
Median	0	Each gate	0.5				98.8%	0x
			0.05				96.6%	0x
	2	Level 1: 511 Level 2: 85.1	0.5				97.5%	2.77x
			0.05				93.7%	2.77x
	3	Level 1: 511 Level 2: 85.1 Level 3: 14.2	0.5				95.8%	4.76x
			0.05				88.1%	4.76x
Average	0						90.2%	0x
	2						87.1%	2.86x
	3						83.6%	6.39x

7. REFERENCES

[1] A. Chandrakasan, S. Sheng, and R. Brodersen, "Low-power CMOS digital design," *IEEE Journal of Solid-State Circuits*, vol. 27, pp. 473-484, April 1992.

[2] J.M. Rabaey and M. Pedram, *Low Power Design Methodologies*, Kluwer Academic Publishers, 1996, pp 21-64,130-160.

[3] R. Nair, C.L. Berman, P.S. hauge, and E.J. Yoffe, "Generation of performance constraints for layout," *IEEE Transactions on Computer-Aided Design*, pp.860-874, Aug. 1989.

[4] T. Gao, P.M. Vaidya, and C.L. Liu,"A new performance driven placement algorithm," *Proc. of ICCAD*, pp. 44-47, 1991.

[5] H. Youssef and E. Shragowitz, " Timing constraints for correct performance," *Proc. of ICCAD*, pp. 24-27, 1990.

[6] P. Pant, V. De, and A. Chatterjee, "Simultaneous power Supply, threshold voltage, and transistor size optimization for low-power operation of CMOS circuits," *IEEE Trans. On VLSI Systems*, vol. 6, no. 4, pp. 538-545, December 1998.

[7] T. Sakurai and A.R. Newton, "Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas," *IEEE Journal Solid-State Circuits*, vol. 25, pp. 584-594, Apr. 1990.

[8] A. Bhavnagarwala, V. De, B. Austin, and J. Meindl, "Circuit techniques for CMOS low power GSI," in Proc. Int. Symp. Low Power Electron. Design: Dig. Tech. Papers, Aug. 1996, pp. 193-196.

[9] A.Raghunathan, N.K. Jha, and S. Dey, *High-Level Power Analysis and Optimization,* Kluwer Academic Publishers, 1998, pp 1-25.

[10] K. Roy and S.C. Prasad, *Low-Power CMOS VLSI Circuit Design,* John wiley & Sons, Inc., 2000, pp. 201-252.

[11] T. Kobayashi and T. Sakurai, "Self adjusting threshold voltage scheme (SATS) for low voltage high speed operation," *IEEE CICC*, 1994, pp.271-277.

[12] S. Mutoh, "1-V Power supply high-speed digital circuit technology with multithreshold-voltage CMOS," *IEEE Journal of Solid-State Circuits*, vol. 30, pp. 847-, April 1992.

[13] A. Fariborz, "A dynamic threshold voltage MOSFET (DTMOS) for ultra-low voltage operation," IEDM Tech., 1994, pp.809-818.

[14] L. Wei, Z. Chen, and K.Roy, "Double gate dynamic threshold voltage (DGDT) SOI MOSFETs for low power high performance designs," IEEE SOI conference, 1997, pp. 82-83.

[15] S.S. Sapatnekar, V.B. Rao, P.M. Vaidya, and S Kang, "An exact solution to the transistor sizing problem ofr CMOS circuits using convex optimization," *IEEE Trans. On CAD of Integrated Circuits and Systems*, vol. 12, no. 11, pp. 1621-1634, September 1993.