

# Total Power Optimization through Simultaneously Multiple- $V_{DD}$ Multiple- $V_{TH}$ Assignment and Device Sizing with Stack Forcing

W. Hung, Y. Xie, N. Vijaykrishnan, M. Kandemir, M. J. Irwin and Y. Tsai

Embedded & Mobile computing Design Center

The Pennsylvania State University

University Park, PA 16802, USA

{whung,yuanxie,vijay,kandemir,mji,ytsai@cse.psu.edu}

## ABSTRACT

In this paper, we present an algorithm for the minimization of total power consumption via multiple  $V_{DD}$  assignment, multiple  $V_{TH}$  assignment, device sizing and stack forcing, while maintaining performance requirements. These four power reduction techniques are properly encoded in genetic algorithm and evaluated simultaneously. The overhead imposed by the insertion of level converters is also taken into account. The effectiveness of each power reduction mechanism is verified, as are the combinations of different approaches. Experimental results are given for a number of 65 nm benchmark circuits that span typical circuit topologies, including inverter chains, SRAM decoders, multiplier and a 32bit carry adders. From the experimental results, we show that the combination of four low power techniques is the effective way to achieve low power budget.

## Categories and Subject Descriptors

C.4.4 [Performance of Systems]: Modeling techniques.

## General Terms

Algorithms, Performance, Design.

## Keywords

Genetic Algorithm, Low Power.

## 1. INTRODUCTION

Process scaling and aggressive performance improvements have resulted in power consumption becoming a first-order design criterion. For example, the latest Intel Pentium 4 processor (Prescott, 2004) has a power consumption of 103 Watts, almost four times larger than that of the Pentium III (1999). In addition to its clear impact on battery lifetime in portable embedded systems, processor power consumption has also become a primary constraint on workstation performance. Reducing power dissipation is a top priority in modern VLSI design.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISLPED'04, August 9–11, 2004, Newport Beach, California, USA.

Copyright 2004 ACM 1-58113-929-2/04/0008...\$5.00.

Power dissipation in CMOS digital circuits consists of dynamic power, short circuit power and static power. Short circuit power consumption can be kept within bounds by careful design and tuning the switching characteristics of complementary logic (slope engineering); it is usually negligible compared to dynamic power and leakage power. Therefore, we will focus on the latter two sources of power consumption, as indicated by equation (1).

In this equation, the first term is the dynamic power dissipation and the second term models the static power dissipation due to leakage current  $I_{leak}$ . ( $A$  is the switching activity factor for dynamic power,  $C$  is the switched capacitance, and  $V_{DD}$  is the supply voltage). Dynamic power was once the dominant power consumption term. However, as the result of technology scaling and  $V_{TH}$  (threshold voltage) decreasing, leakage power will soon account for a large portion of total power consumption.

$$P \approx ACV_{DD}^2 \cdot f + I_{Leak} \cdot V_{DD} \quad (1)$$

Although there are many techniques to reduce power dissipation, most existing works focus on one technique in isolation instead of concurrently applying a number of power minimization techniques. In this paper, we propose a power optimization framework based on the genetic algorithm. The optimization strategy combines four power reduction techniques: multiple  $V_{DD}$  assignment, multiple  $V_{TH}$  assignment, gate sizing, and stack forcing. It simultaneously applies and evaluates the effects of these techniques to achieve maximum power saving under a hard timing constraint. The framework can be easily extended to include other power reduction techniques. To the best of our knowledge, this is the first power optimization framework that simultaneously uses all of these four power reduction techniques.

The paper is organized as follows. Section 2 reviews different power reduction techniques and previous works. Section 3 gives a brief introduction on genetic algorithm. Section 4 describes our power optimization framework based on genetic algorithm. Experimental results are presented and explained in Section 5. Section 6 presents conclusions and future work.

## 2. POWER REDUCTION TECHNIQUES AND RELATED WORKS

Due to the quadratic relationship between dynamic power consumption and  $V_{DD}$ , reducing the supply voltage is the most effective way to lower the dynamic power, at the expense of increasing gate delay. In order to prevent the negative effect on

performance, the threshold voltage ( $V_{TH}$ ) must be reduced proportionally with the supply voltage so that a sufficient driving current is maintained. This reduction in the threshold voltage causes an exponential increasing in leakage power, which in turn can raise the static power of the device to unacceptable levels.

To counter the loss in performance while improving the power efficiency, **multiple  $V_{DD}$**  [1] and **multiple  $V_{TH}$**  [2] techniques have been proposed. The gates on critical paths operate at the higher Vdd or lower Vth, while those on non-critical paths operate at the lower Vdd or higher Vth, thereby reducing overall power consumption without performance degradation. These techniques have been successfully implemented. For example, IBM's ASIC design flow can fully take advantage of the power-performance tradeoff by using their voltage island concept and multiple-Vth standard cell library [3]. **Gate sizing** [4] is another powerful method of power optimizing. Logic gates on critical paths may be sized up to meet timing requirement, at the expense of higher power consumption; while those on non-critical paths can be sized down to reduce the power consumption. Hamada *et al.* [5] examined multiple supply voltages, multiple threshold voltages and transistor sizing individually and derived a set of rules of thumb for optimal supply voltages, threshold voltages and transistor sizing. A good summary of these three techniques is presented by Brodersen *et al* [6].

To tackle the ever-increasing leakage power, besides multiple Vth technique, another solution is **stack forcing**. It has been shown that the stacking of two off transistors can significantly reduce leakage power than a single off transistor [7]. Therefore, we can force a non-stack device to a stack of N devices without affecting the input load. Figure 1 shows a stacking force example for an inverter when  $N=2$ . By ensuring the input load unchanged, we guarantee that the previous stages' delay and dynamic power consumption are not affected. The logic gate with stack forcing has much lower leakage power, however, at the expense of a delay penalty, because the effective device width  $W_{eff}$  becomes  $W/N^2$  after stack forcing. It is similar to replacing a low-Vt device with a high-Vt device in a multiple-Vt design.

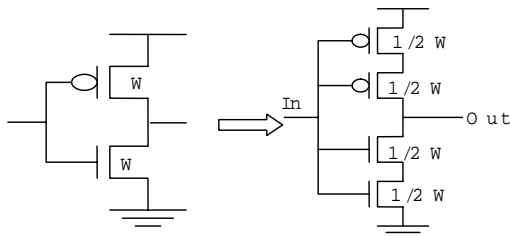


Figure 1. Stacking force example for inverter.

To achieve the most power efficient design, all these power reduction techniques have to be balanced. Stojanovic *et.al.* [8] combined gate sizing and supply voltage optimization to minimize power consumption under a delay constraint. Roy *et al.* [9] presented a heuristic algorithm to combine dual-Vdd and dual-Vth techniques. Augsburger [10] evaluated the effectiveness of multiple supply voltage, transistor sizing, and multiple thresholds independently and in conjunction with each other, showing that the order of application of these techniques determines the final savings in active and leakage power.

Recently, researchers have looked at the joint optimization of these techniques, because it can help to achieve maximum power savings compared to a sequential application of a single variable optimization. Sirichotiyakul *et al* [11] presented an algorithm for

joint optimization of dual-vt and sizing to reduce leakage power. Karnik *et al* [12] developed a heuristic iterative algorithm to do device sizing and dual-Vt allocation simultaneously to exploit the timing slack for reduction of total power consumption. They found that joint dual-vt and sizing can reduce the power by 10% and 25% compared with pure Vt allocation or pure sizing method, respectively. Srivastava *et al.* [13] were the first to investigate the effectiveness of simultaneously multiple supply and threshold voltage assignment for total power saving. Their algorithm is based on linear programming approach. Nguyen *et al.* [14] developed another linear programming algorithm that can simultaneously perform the threshold voltage assignment and sizing optimization, and then apply the supply voltage optimization as a sequential step. Lee *et al.* [19] proposed heuristic algorithms for simultaneous state, Vth and gate oxide assignment. Srivastava *et al.* [20] proposed a sensitivity-based algorithm to perform concurrent sizing, Vdd and Vth assignment.

In this paper, we present a GA-based power optimization framework that can simultaneously exploit four power optimization techniques: multiple supply voltage assignment, multiple threshold voltage assignment, gate sizing and force stacking. To the best of our knowledge, none of the previous works can do simultaneously joint optimization of more than three techniques, and the framework can easily be extended to include other techniques such as multiple gate oxide assignment.

### 3. GENETIC ALGORITHM

Genetic algorithms (GA) [15] are a class of search and optimization methods that mimic the evolutionary principles in natural selection. The solution is usually encoded into a binary string called *chromosome*. Instead of working with a single solution, the search begins with a random set of chromosomes called *initial population*. Each chromosome is assigned a *fitness* that is directly related to the *objective function* of the optimization problem. The population of chromosomes is modified to a new *generation* by applying three *operators* similar to natural selection operators – *reproduction*, *crossover* and *mutation*. Reproduction selects good chromosomes based on the fitness function and duplicates them. Crossover picks two chromosomes randomly and some portions of the chromosomes are exchanged with a probability  $P_c$ . Finally, mutation operator changes a 1 to a 0 and vice versa with a small mutation probability  $P_m$ . A genetic algorithm successively applies these three operators in each generation until a termination criterion is met. It can very effectively search a large solution space while ignoring regions of the space that are not useful. This algorithmic methodology leads to very time-efficient searches. In general, a genetic algorithm has the following steps:

1. Generation of initial population.
2. Fitness function evaluation.
3. Selection of chromosome.
4. Reproduction, Crossover and Mutation operations.

### 4. POWER OPTIMIZATION FRAMEWORK

Our power optimization flow uses a genetic algorithm and is shown in Figure 2. The circuit configuration information, such as

supply voltage assignment and gate sizing, are encoded into binary strings called chromosome. The optimization flow begins with a random generated *initial population*, which consists of many randomly generated circuit configurations. The optimization flow is an iterative procedure. The chromosomes with better fitness will survive at each generation and are applied three different operations (reproduction, crossover and mutation) to be a new set of chromosomes – or new circuit configuration. The iteration continues until the termination criterion is met.

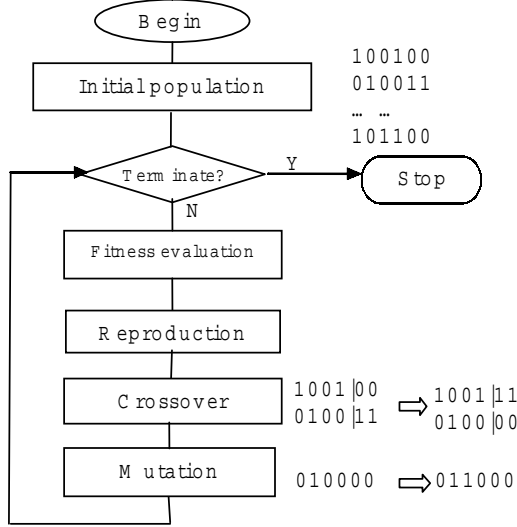


Figure 2. The power optimization flow.

## 4.1 Chromosome encoding

Given different power reduction techniques, we can encode all the tuning variables into a binary chromosome string. Figure 3 shows both the structure of a chromosome and an encoding example, representing  $N$  gates in a circuit. This encoding example is based on the assumption that we use a dual- $V_{dd}$ , dual- $V_{th}$  library with four discrete sizes for each type of gate, and the gate has one forced stacking version. While the  $V_{th}$  allocation and force stacking can be done in transistor level, for simplicity we assume the granularity is at the gate level. For example, in Figure 3, the chromosome shows that Gate 1 is assigned to use lower  $V_{dd}$ , higher  $V_{th}$ , size 3 and no force stacking. Note that only the tuning variables are encoded into the chromosome. The type of each logic gate and the circuit topology information are known *a priori* to calculate the power and delay based on the chromosome configuration.

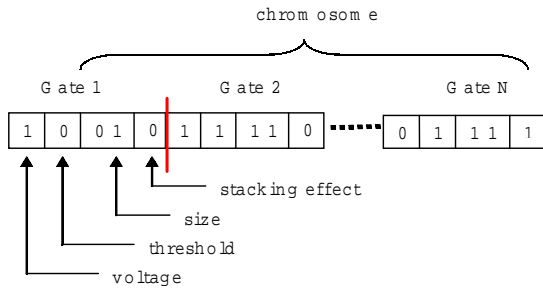


Figure 3. The structure of a chromosome.

This encoding scheme is easy to extend for more complicated standard cell library. For example, if a standard cell library has more than two supply voltage choices or more than two threshold voltage choices, we just need more bits in the chromosome. Increasing the flexibility of the library simply increases the bits required for each gate. Since the chromosomes are randomly generated and we are using binary bit string for chromosome, in some case, the chromosome configuration may not be valid. For example, for a library with three threshold voltage choices, we have to use two bits for the  $V_{th}$  configuration and there will be one invalid configuration. We tackle this problem using a penalty function, as described in section 4.2.

## 4.2 Fitness function

The fitness function, which decides the surviving chance for a specific chromosome, is related to the power consumption and the delay of the circuit, as well as the validity of the chromosome.

### 4.2.1 Power

In our power optimization framework, the goal is to find a configuration such that the power consumption for the circuit is as low as possible. Therefore the fitness of a chromosome should be related to the power consumption of that particular configuration, which can be calculated using equation (1) discussed in section 1. For the dynamic power of the gate, the switching activity is obtained by exhaustive simulation with the assumption that the input probabilities of being high or low are equal and independent. The gate leakage and sub-threshold leakage were also characterized by similar simulation. Our method is general enough and more accurate power estimates can be used, if more information on probabilities is available.

When using multiple supply voltages in a circuit, level converters are required whenever a logic gate at the lower supply has to drive a gate at the higher voltage [4]. The overall power consumption for the circuit should also include the level converters' power consumptions.

### 4.2.2 Delay

It should be noted that the power optimization is under a specified timing constraint. If the critical path delay in a circuit is longer than the timing requirement, the configuration is not desirable and the corresponding chromosome should have little chance to survive. The delay calculation of the circuit is based on logical effort [16]. Again, the delay from level converters is taken into account.

### 4.2.3 Validity

As we have discussed in section 4.1, since we use binary string to represent a chromosome, when the number of choices for a tuning variable is not  $2^N$  (for example, a gate has six discrete sizes, or three threshold voltage choices), we may end up with an invalid configuration during the population initialization or chromosome operations. For those chromosomes representing invalid configuration, the chance of surviving should also be very small. Based on the above argument, the fitness function can be defined as

$$Fitness = \frac{1}{Total\_power} - Penalty \quad (2)$$

where the penalty is a big number if timing violates or the chromosome is invalid, such that those valid chromosomes with lower power and meet timing requirement have better fitness to survive.

### 4.3 Control parameters

While generating the initial population, we have to set an appropriate population size, and the crossover probability  $P_c$ , as well as the mutation probability  $P_m$ . If the population size is too small, the genetic diversity within the population may not increase for many generations. On the other hand, a large population size increases the computation time for each generation but it may take fewer generations to find the best solution. Schaffer *et al.* [18] have conducted extensive simulation on a wide range of functions and concluded that a small population of size 20 to 30, a crossover probability in the range of 0.75 to 0.95, and a mutation probability in the range of 0.005 to 0.01 perform very well. In our implementation, we set the population size to be 100, crossover probability  $P_c$  to be 0.9 and the mutation probability  $P_m$  to be 0.01.

The termination of the iterative evolution can be user defined. We set a maximum generation and specify that if the power reduction is less than 0.001% during the last 100 generations, the evolution stops without going through all generations.

## 5. EXPERIMENTAL RESULTS

We implement our algorithm in C. To test our algorithm, we construct a dual-V<sub>dd</sub> and dual-V<sub>th</sub> standard cell library using 65 nm process. The logic gates in the library are inverter, NAND2, NOR2, XOR2 and a level converter. The level converter implementation [4] is shown in Figure 4.

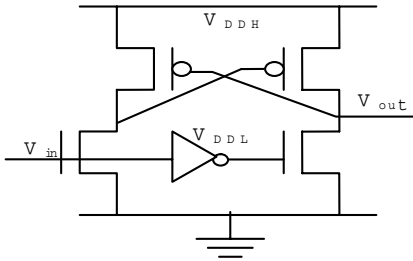


Figure 4. The implementation of the level converter.

Both V<sub>th</sub> allocation and force stacking can be done at a transistor level and the gates may have several discrete sizes. However, for simplicity, we assume the granularity of V<sub>th</sub> assignment and forced stacking are at the gate level and the gates have only two discrete sizes, such that we need only 4 bits to encode the tuning variables. Increasing the flexibility only increases number of bits to represent each gate's configuration. Based on the conclusion in [13] that the optimal second V<sub>dd</sub> in dual-V<sub>th</sub> system should be ~50% of the higher supply voltage, the possible gate supply voltages for our library are 1V, 0.5V. For NMOS (PMOS) transistors, the high threshold voltage and the low threshold voltage are 0.22V (-0.22V) and 0.12V (-0.12V) respectively. The library was characterized using Berkeley 65 nm BSIM predictive model [17]. The gate leakage and sub-threshold leakage were pre-characterized.

The benchmark circuits we choose to map to our library span typical circuit topologies, including a circuit from ISCAS 85 benchmark, an inverter chain, a 32-bit carry-ripple adder, an 8x8 carry-save multiplier, an 8-to-256 SRAM decoder and three manually generated circuits. The circuit sizes range from 8 gates to 1050 gates. The circuits were first optimized for maximum speed (e.g. using all higher V<sub>dd</sub> and lower V<sub>th</sub> on the critical path) and then were optimized for lowest power consumption. We then perform power optimization with the timing constraint relaxation percentage. The runtime for our benchmark on an Intel Pentium 4 processor (2.8 GHz 512M RAM) ranges from 0.57s to 497s, depending on circuit sizes and timing constraints.

Figures 5 and 6 present the static power and dynamic power breakdown for maximal speed optimized circuit and minimal power optimized circuit respectively. With our 65nm library, the static power accounts for average 74% of the total power while the dynamic power accounts for 26% of the overall power. After applying all four power reduction techniques without any timing constraint circuits, the static power accounts for only about 20% of the overall power while dynamic power accounts for 80%. A significant part of overall power reduction are from the static power reduction by using these four techniques together. Overall 84% of the power reduction was from the static power reduction.

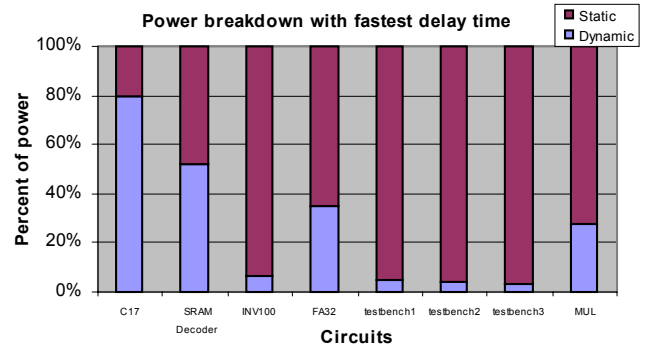


Figure 5. Power breakdown for speed-optimized circuits.

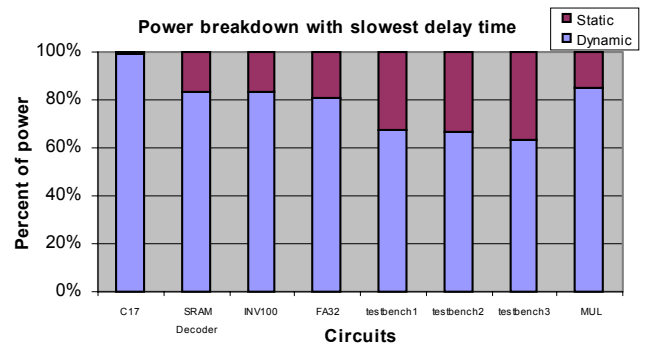
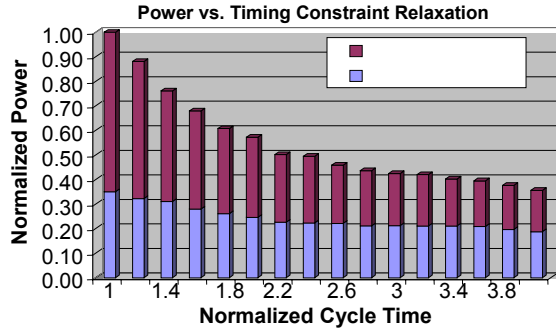


Figure 6. Power breakdown for power-optimized circuits.

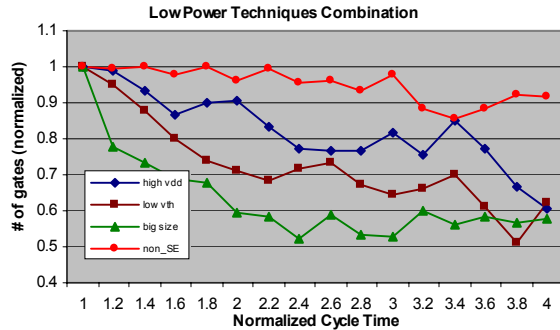
Figure 7 shows the change in power consumption with the application of our algorithm. The timing constraint (cycle time) and power consumption are both normalized to the fastest speed case. This plot is for 32-bit carry-ripple adder. We can see that when the timing constraint is relaxed to be twice long as the fastest speed, the power reduction we can achieve is about 45%. Further relaxation we can achieve more than 65% when the

requirement is 4 times as long as the fastest critical delay. We can also see that most of the power reduction comes from the static power reduction.



**Figure 7. Power reduction as timing constraint is relaxed.**

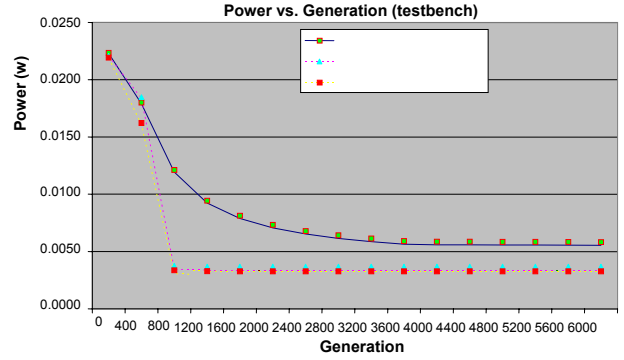
Figure 8 shows the power reduction techniques that are used in the circuits when the timing constraint is relaxed. The number of gates using a specific configuration is normalized to the maximum-speed optimized case. It is observed that as the timing constraint relaxed from 0% to 400%, the gates tend to use the less power consuming configuration, i.e., lower Vdd, higher Vth, smaller size and force stacking. The interaction of these four tuning variables (Vdd, Vth, sizing and force stacking) can also be observed from our experiments, which are represented by the fluctuations of four curves.



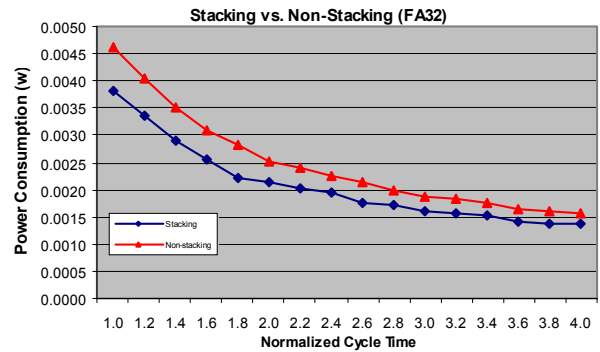
**Figure 8. Low Power Techniques Combinations.**

Figure 9 shows the power reduction process for the testbench circuit with 1050 gates under different timing relaxation. We can see that for very tight timing constraint (1% timing relaxation), the algorithm achieves a maximum power saving within 3000-4000 generation under the timing constraint. Running more generations won't help much. If we relax the timing constraint, the convergence can be achieved much faster (within 1000 generations).

Figure 10 shows the power consumptions between configuration with stacking force and without stacking force. The comparison is conducted with all four low power techniques available and all techniques except stacking force. As we can see from the picture, the gap between two curves gets closer when the timing constraint is relaxed more. This means that when timing constraints is tight, the configuration without stacking force will consume more static power. With relaxed timing constraints, there is more flexibility in obtaining low power solution close to the best even without stacking

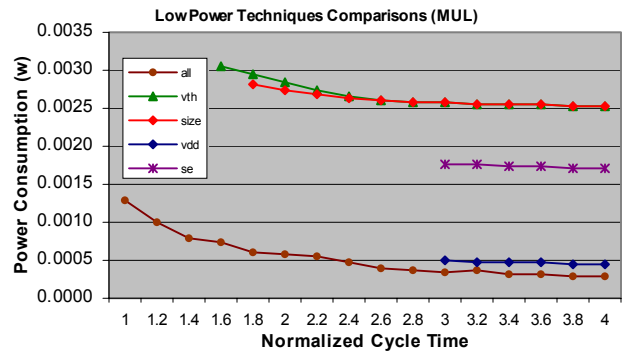


**Figure 9. Power reduction over entire run.**



**Figure 10. Comparison of power consumption with/without stacking force.**

Figure 11 shows the comparison of power consumption with only one low power technique available at a time to one equipped with four low power techniques. The reason for some curves not showing the point from the beginning is that they simply can not meet timing requests at that point. We can easily see from the graph that the dual Vdd is the most effective way to reduce power while the dual Vth is the less effective one. But dual Vth can still maintain good timing constraint achievements compared to the other three techniques. The curve with all available techniques can easily maintain low power consumption without incurring performance degradation.



**Figure 11. Different Technique Comparisons.**

## Comparison to ILP approach

Compared to the ILP approach that were used by [13] [14], one advantage of our GA approach is that the parallel nature of genetic algorithms suggests parallel processing as the natural route to explore. We implement a parallel version of our algorithm, by dividing the population processing among multiple processors. We notice that in average more than 3X run-time speed-up on a 4-processor workstation against the single-processor version of our algorithm (The reason that we cannot achieve a 4X speedup is the interaction overhead among parallel processes). Another advantage is that for ILP approach, the running time for a large circuit may be prohibitively long; while for our GA-based strategy, we can set a proper termination criterion to tradeoff the runtime and power saving.

## 6. CONCLUSION

We present a power optimization framework based on genetic algorithm. The optimization strategy can simultaneously perform multiple-Vdd assignment, multiple-Vth assignment, gate sizing in conjunction with stack forcing technique to minimize total power consumption, while maintaining performance requirements. The framework can be easily extended to include other power reduction techniques, such as multiple gate oxide [19].

## 7. ACKNOWLEDGEMENT

The authors would like to thank Prof. Robert Dick in Northwestern University for valuable comments.

## REFERENCES

- [1] K. Usami and M. Horowitz, "Clustered voltage scaling techniques for low-power Design," ISLPED 1995.
- [2] Q. Wang and S.Vrudhula, "Algorithms for minimizing standby power in deep submicron, dual-Vt CMOS circuits," IEEE Transactions on CAD, vol.21, p.306-318, 2002.
- [3] R. Puri, L. Stok, J. Cohn, D. Kung, D. Pan, D. Sylvester, A. Srivastava, and S.H. Kulkarni, "Pushing ASIC performance in a power envelope," ACM/IEEE Design Automation Conference, pp. 788-793, 2003.
- [4] J. Rabaey, A. Chandrakasan and B. Nikolic, *Digital Integrated Circuits: A Design Perspective*, second edition, Prentice Hall, NJ, 2003.
- [5] M. Hamada, Y. Ootaguro, "Utilizing Surplus Timing for Power Reduction," Proc. of the IEEE Custom Integrated Circuits Conference 2001, pp. 89-92.
- [6] R. Brodersen, M. Horowitz, D. Markovic, B. Nikolic, and V. Stojanovic, "Methods for True Power Minimization," Proc. ICCAD, San Jose, CA, November 2002. pp 35-42
- [7] S. Narendra, S.Borkar, V.De, D.Antoniadis and A.Chandrakasan, "Scaling of Stack Effect and its Application for Leakage Reduction," IEEE Proc. of Low Power Electronics and Design, pp.195-200, 2001.
- [8] V. Stojanovic, D. Markovic, B. Nikolic, M. Horowitz, and R. Brodersen, "Energy-Delay Tradeoffs in Combinational Logic Using Gate Sizing and Supply Voltage Optimization," Proc. European Solid-State Circuits Conf., Florence, Italy, September 2002.
- [9] K. Roy, L. Wei, and Z. Chen, "Multiple Vdd Multiple Vth (MVC MOS) for Lower Power Applications," International Symposium on Circuits and Systems, Vol 1, pp. 366-370, 1999.
- [10] S. Augsburger, B. Nikolic., "Reducing Power with Dual Supply, Dual Threshold and Transistor Sizing," International Conference on Computer Design, ICCD'02, pp.316-321, September, 2002.
- [11] S. Sirichotiyakul, T. Edwards, C. Oh, J. Zuo, A. Dharchoudhury, R. Panda, and D. Blaauw., "Stand-by Power Minimization through Simultaneous Threshold Voltage Selection and Circuit Sizing," Proc. ACM/IEEE Design Automation Conf., pages 436--441, 1999.
- [12] T.Karnik, Y.Ye, J. Tschanz, L. Wei, S.Burns, V. Govindarajulu, V.De and S.Borkar, "Total Power Optimization by Simultaneous Dual-Vt Allocation and Device Sizing in High Performance Microprocessors," Proc. Design Automation Conference, 2002, pp.486
- [13] Srivastava and D. Sylvester, "Minimizing total power by simultaneous Vdd/Vth assignment," IEEE/ACM Asia-South Pacific Design Automation Conference, pp. 400-403, 2003.
- [14] D. Nguyen, A. Davare, M. Orshansky, D. Chinnery, B. Thompson, K. Keutzer, "Minimization of Dynamic and Static Power Through Joint Assignment of Threshold Voltages and Sizing Optimization," ISLPED 2003.
- [15] Goldberg, D.E. *Genetic Algorithms in Search, Optimization, and Machine Learning*, New York: Addison-Wesley. 1989.
- [16] Sutherland, Sproull and Harris, *Logical effort*, 1999.
- [17] <http://www.device.eecs.berkeley.edu>, Berkeley predictive model.
- [18] J.Schaffer, J. Caruana, L. Eshelman and R.Das, "A study of control parameters affecting online performance of genetic algorithms for function optimization," Proc. of the Third international Conference on Genetic Algorithms, pp.51-60
- [19] D.Lee, H.Deogun, D.Blaauw, D.Sylvester, "Simultaneous State, Vt and Tox Assignment for Total Standby Power Minimization," Proc. of DATE 2004. pp.494-499
- [20] Srivastava and D. Sylvester, D.Blaauw, "Concurrent Sizing, Vdd and Vth Assignment for Low-Power Design," Proc. of DATE 2004, pp.718-719