A NOVEL DYNAMIC POWER CUTOFF TECHNOLOGY (DPCT) FOR ACTIVE LEAKAGE REDUCTION IN DEEP SUBMICRON VLSI CMOS CIRCUITS

BY BAOZHEN YU

A dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Prof. Michael L. Bushnell

and approved by

_____

_____

_____

_____

_____

New Brunswick, New Jersey

October, 2007

ABSTRACT OF THE DISSERTATION

A Novel Dynamic Power Cutoff Technology (DPCT) for Active Leakage Reduction in Deep Submicron VLSI CMOS Circuits

by Baozhen Yu

Dissertation Director: Prof. Michael L. Bushnell

Due to the exponential increase of subthreshold and gate leakage currents with technology scaling, leakage power is increasingly significant in CMOS circuits as the technology scales down. The leakage power is as much as 50% of the total power in the 90*nm* technology and is becoming dominant in more advanced CMOS technologies with smaller feature sizes. Also, the leakage in active mode is significantly larger due to the higher die temperature in active mode. Although many leakage reduction techniques have been proposed, most of them can only reduce the circuit leakage power in standby mode.

In this thesis, we present a novel active leakage power reduction technique using dynamic power cutoff, called the *dynamic power cutoff technique* (DPCT). To reduce the active leakage power, we target the idle part of the circuit when it is in active mode. First, the *switching window* for each gate, during which a gate makes its transitions, is identified by static timing analysis. Then, the circuit is optimally partitioned into different groups based on the *minimal switching window* (MSW) of each gate. Finally, power cutoff transistors are inserted into each group to control the power connections of that group. The power of each gate is only turned on during a small timing window within each clock cycle, which results in significant active leakage power savings. Standby leakage can also be reduced by turning off the power connections of all gates all of the time once the circuit is idle. This technique also reduces dynamic power and short-circuit power by reducing the circuit glitches.

Experimental results on *ISCAS '85* benchmark circuits at the logic level modeled using

70*nm* Berkeley Predictive Models show up to 90% of active leakage, 99% of standby leakage, up to 54% of dynamic, and up to 72% of total power savings. DPCT can also reduce the maximal voltage drop on the power grid by more than 30% on average. With process variations, the average total power and active leakage power savings will be reduced by 12.7% and 14.8%, respectively. In spite of that, DPCT still gives excellent power savings, which are 73.6% of active leakage power and 34.7% of total power under process variations. We also implemented the layouts of a 16-bit multiplier and a c432 using DPCT. The experimental results for the layout designs confirmed the effectiveness of DPCT in physical level design.

Acknowledgements

First, I would like to thank Prof. Michael Bushnell, my advisor, for his guidance, encouragement and patience on my Ph.D. research and dissertation. Second, I would like to thank all my committee members for their comments on my dissertation. Finally, I would like to thank all of the members of my research group for their valuable comments and suggestions on my dissertation.

Dedication

To my grandmother, father, mother, sister, and my wife.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background and Motivation

There are three sources of power dissipation in CMOS digital circuits: dynamic power, short-circuit power and leakage power. Usually, the dynamic power is dominant and the other two parts are negligible. But this will not be the case as the CMOS technology scales down further. As the CMOS technology scales down, the supply voltage must be reduced such that dynamic power can be kept at reasonable levels. In order to prevent a negative effect on performance, the threshold voltage must be reduced at a rate such that a sufficient gate overdrive is maintained. This reduction in the threshold voltage causes an increase in the leakage current of about 5 times per generation, which in turn can increase the static power of the device to unacceptable levels. In 0.1um CMOS technology, the leakage power is approaching 30% of the total processor power [13]. The leakage power is as much as 50% of the total power in the 90nm technology [23]. Thus, leakage reduction is necessary for CMOS technologies below 0.1um. Also, leakage is important in both standby and active operation modes. Actually, the leakage in active mode is significantly larger due to the higher die temperature in active mode.

To solve the leakage problem many leakage reduction techniques have been proposed. Among them, some require modification of the process technology, achieving leakage reduction during the fabrication stage. Others are based on circuit-level optimization schemes that require architecture support. In spite of all these available techniques to reduce leakage power in circuits, leakage power still remains a big problem for deep submicron circuits. Furthermore, most of the available leakage reduction techniques can only reduce the circuit leakage power in standby mode. So, more efficient active leakage power reduction techniques are still necessary to keep the leakage power under control as CMOS technology scales down.

1.2 Problem Statement

The problem to be solved in this work is: Given a CMOS circuit, find a technique to reduce the active leakage power in the circuit significantly without introducing much performance cost. The implementation complexity of the technique should be feasible so that it can be practical for large circuits.

1.3 Summary of Original Contributions

In this work, we present a novel active leakage power reduction technique using dynamic power cutoff, called the *dynamic power cutoff technique* (DPCT). We propose a new *minimal switching window* (MSW) for CMOS gates to identify when the gate is active, which is equal to the worst-case delay of the gate. We propose a heuristic partitioning algorithm based on dynamic program to partition the circuit into groups based on the MSW of each gate so that the cost of adding extra power cutoff controls will be minimized without sacrificing much of the leakage power savings. We propose a six-step approach to implement DPCT. We also present the procedures to do power grid analysis and process variation analysis on DPCT.

Experimental results on *ISCAS '85* benchmark circuits modeled using 70*nm* Berkeley Predictive Models [17] show up to 90% in active leakage power saving, 99% in standby leakage saving, up to 54% in dynamic power saving, and up to 72% in total power saving. DPCT can also reduce the maximal voltage drop on the power grid by more than 30% on average. With process variations, the average total power and active leakage power savings will be reduced by 12.7% and 14.8%, respectively. In spite of that, DPCT still gives excellent power savings, which are 73.6% of active leakage power and 34.7% of total power with process variations. We also implemented the layouts of a 16-bit multiplier and c432 using DPCT. The 16-bit multiplier with DPCT saves 54.7% of the total power, 85.7% of the active leakage power (including short-circuit power) and 38.1% of the dynamic power with 7.7% delay overhead and 8.6% area overhead. The c432 circuit with DPCT saves 22.5% of the total power, 73.6% of the active leakage power (including short-circuit power) and 2.3% of the dynamic power with 9% delay overhead and 13.7% area overhead. The experimental results on the layout designs confirmed the effectiveness of DPCT in physical level design.

1.4 Organization of the Thesis

In Chapter 2, we introduce some background on the power dissipation of CMOS circuits

and present a survey on prior leakage reduction techniques. In Chapter 3, we introduce the background and prior work on the power grid analysis and the process variation analysis. The detail of our novel DPCT technique is introduced in Chapter 4. The procedures to analyze the power grid and process variations for DPCT are presented in Chapter 5. Chapter 6 presents the experimental results on the power savings, power grid analysis and process variation analysis for DPCT. Chapter 7 presents a layout implementation of 16-bit multiplier with DPCT. Chapter 8 gives the future work and concludes.

Chapter 2

Prior Work: Techniques for Leakage Power Reduction

2.1 Introduction

In this section, we first review the basic mechanisms of power dissipation in CMOS circuits, where we focus on the mechanisms of leakage power. Then we review some existing leakage reduction techniques.

2.2 Power Dissipation in CMOS Circuits

There are three sources of power dissipation in CMOS digital circuits: *dynamic* power, *short-circuit* power, and *leakage* power. Formerly, the dynamic power was dominant and the other two parts were negligible. But leakage power is becoming more and more significant as the CMOS technology goes into the deep submicron scale. Now, all three are important and leakage power is beginning to dominate.

2.2.1 Dynamic Power

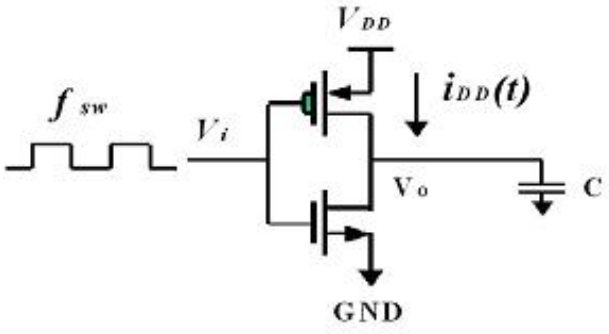


Figure 2.1: A CMOS Inverter

Dynamic power is the power required to charge and discharge the load capacitances when

transistors switch. Suppose that we have a CMOS inverter with load capacitance C, which is shown in Figure 2.1. One cycle involves a rising and a falling transition at the gate output. Charge $Q = CV_{DD}$ is required on a low-to-high transition at the gate output and the charge is dumped to *GND* during the high-to-low transition at the gate output. This charging and discharging process repeats $T f_{sw}$ times over an interval of T, where f_{sw} is the frequency of the input signal. So, the dynamic power can be calculated by the following formula:

$$P_{dynamic} = \frac{1}{T}\int_0^T i_{DD}(t)V_{DD}\,dt = \frac{V_{DD}}{T}\int_0^T i_{DD}(t)\,dt = \frac{V_{DD}}{T}(T f_{sw} C V_{DD}) = C V_{DD}^2 f_{sw} \tag{2.1}$$

The dynamic power can also be formalized as: $P_{dynamic} = \alpha C_L V_{DD}^2 f$, where f is the clock frequency and α is the node transition activity factor.

2.2.2 Short-Circuit Power

When transistors switch, both *n*MOS and *p*MOS networks may be momentarily on at once. This leads to a blip of short circuit current. The short circuit power is given by:

$$P_{short-circuit} = I_{mean} V_{DD} \tag{2.2}$$

where I_{mean} is average short-circuit current. For a symmetric inverter shown in Figure 2.1,

$$I_{mean} = \frac{\beta}{12}(V_{DD} - 2V_t)^3 \frac{t_{rf}}{t_p} \tag{2.3}$$

where V_{DD} is the power supply voltage, $V_t = V_{tn} = -V_{tp}$ is the threshold of the MOSFETs, $\beta = \beta_n = \beta_p$ is the β of the MOSFETs, $t_r = t_f = t_{rf}$ are the rising and falling times of the input pulse, and t_p is the period of the input pulse [81].

2.2.3 Leakage Power

Leakage power, also called static power, is due to the off-state current of a transistor when it is off. Suppose that there are N transistors in a circuit, and I_{off_i} is the off-state current of the ith transistor. Then, the total leakage power of the circuit can be expressed in the following formula:

$$P_{leakage} = V_{DD} \sum_{i=1}^{N} I_{off_i} \tag{2.4}$$

There are mainly six short-channel leakage mechanisms as illustrated in Figure 2.2 [34]. I_1 is the reverse-bias *pn* junction leakage; I_2 is the subthreshold leakage; I_3 is the oxide tunneling

current; I_4 is the gate current due to hot-carrier injection; I_5 is the *gate-induced drain leakage* (GIDL); and I_6 is the channel punchthrough current. Currents I_2, I_5, and I_6 are off-state leakage mechanisms, while I_1 and I_3 occur in both ON and OFF states. I_4 can occur in the off state, but more typically occurs when the transistor bias states are in transition.

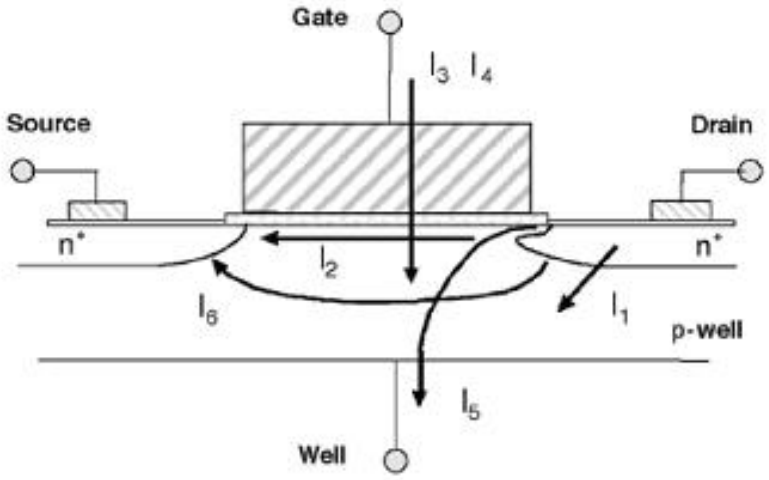


Figure 2.2: Leakage Current Mechanisms of Deep-Submicron Transistors [34]

2.2.3.1 *pn* Junction Reversed-Bias Current

Drain and source to well junctions are typically reverse biased, causing *pn* junction leakage current. The *pn* junction reverse-bias leakage is a function of junction area and doping concentration [61]. If both *n* and *p* regions are heavily doped (this is the case for advanced MOSFETs using heavily doped shallow junctions and halo doping for better short channel effects (SCEs)), *band-to-band tunneling* (BTBT) dominates the *pn* junction leakage. The tunneling current density is given by [76]:

$$J_{b2b} = A\frac{EV_{app}}{\sqrt{E_g}}exp(-B\frac{E_g^{\frac{3}{2}}}{E}),\ A = \frac{\sqrt{2m^*}q^3}{4\pi^3h^2},\ and\ B = \frac{\sqrt{2m^*}}{3qh} \tag{2.5}$$

where m^* is effective mass of the electron; E_g is the energy band gap; V_{app} is the applied reverse bias; E is the electric field at the junction; q is the electronic charge; and h is Planck's constant.

2.2.3.2 Subthreshold Leakage Current

Subthreshold or weak inversion conduction current between source and drain in a MOS transistor occurs when the gate voltage is below V_{th}. It typically dominates modern device off-state leakage. The weak inversion current can be expressed based on the following [89]:

$$I_{ds} = I_{ds0} e^{\frac{V_{gs}-V_{th}}{nv_T}} (1 - e^{\frac{-V_{ds}}{v_T}}) \tag{2.6}$$

$$I_{ds0} = \beta v_T^2 e^{1.8} \tag{2.7}$$

where V_{th} is the threshold voltage; V_{gs} is gate-source voltage; V_{ds} is drain-source voltage; v_T is the thermal voltage; I_{ds0} is the current at the threshold and is dependent on process and device geometry; the $e^{1.8}$ term was found empirically; and n is a process-dependent term affected by the depletion region characteristics and is typically in the range of 1.4-1.5 for CMOS processes.

The inverse of the slope of the $\log_{10} I_{ds}$ versus V_{gs} characteristic is called the *subthreshold swing* (S_t). Subthreshold slope indicates how effectively the transistor can be turned off (rate of decrease of I_{off}) when V_{gs} is decreased below V_{th}. S_t is given by Equation 2.8, where C_{dm} is the capacitance of the depletion layer, and C_{ox} is the gate oxide capacitance [76].

$$S_t = 2.3 \frac{kT}{q} (1 + \frac{C_{dm}}{C_{ox}}) \tag{2.8}$$

Many factors affect the subthreshold current, such as temperature, body effect, DIBL (*drain induced barrier lowering*), the narrow-width effect, the effect of channel length, and V_{th} rolloff.

Temperature.

Subthreshold leakage increases as temperature is raised due to the change of the two parameters: (1) S_t linearly increases with temperature; and (2) the threshold voltage V_{th} decreases.

Body Effect.

Body effect is due to the change of threshold with the substrate bias voltage, which is given by the equation:

$$V_{th} = V_{fb} + \Psi_B + \frac{\sqrt{2\varepsilon_{si} q N_A (2\Psi_B + V_{bs})}}{C_{ox}} \tag{2.9}$$

where V_{bs} is the substrate bias voltage, V_{fb} is the flat-band voltage, N_A is the doping density in the substrate, C_{ox} is the gate oxide capacitance, ε_{si} is permittivity of silicon, and Ψ_B is the difference between the Fermi potential and the intrinsic potential in the substrate. A change of body bias can change the threshold voltage, which will in turn change the leakage current.

DIBL.

In a short-channel device the source and drain depletion widths in the vertical direction and the source drain potential have a strong effect on the band bending over a significant portion of the device. Therefore, the threshold voltage, and consequently the subthreshold current of short-channel devices, vary with the drain bias. This effect is referred to as DIBL. DIBL does not change the subthreshold slope S, but does lower V_{th}, which in turn will increase the subthreshold current.

Narrow-Width Effect.

The decrease in gate width modulates the threshold voltage of a transistor, and thereby modulates the subthreshold leakage.

Effect of Channel Length and V_{th} Rolloff.

Threshold voltage of a MOSFET decreases as the channel length is reduced. This reduction of the threshold voltage with reduction of channel length is known as V_{th} rolloff. The principal reason behind this effect is the presence of 2-D field patterns in short-channel devices instead of one-dimensional (1-D) field patterns in long-channel devices.

2.2.3.3 Tunneling into and through Gate Oxide

Reduction of gate oxide thickness results in an increase in the field across the oxide. The high electric field coupled with low oxide thickness results in tunneling of electrons from substrate to gate and also from gate to substrate through the gate oxide, resulting in the gate oxide tunneling current. This is becoming a significant part of leakage power consumption. The mechanism of tunneling between substrate and gate polysilicon can be primarily divided into two parts, namely: (1) *Fowler-Nordheim* (FN) tunneling; and (2) direct tunneling. In the case of FN tunneling, electrons tunnel through a triangular potential barrier, whereas in the case of direct tunneling, electrons tunnel through a trapezoidal potential barrier. The current density in the FN tunneling is given by [76]:

$$J_{FN} = \frac{q^3 E_{ox}^2}{16\pi^2 h\phi_{ox}} \exp\left(-\frac{4\sqrt{2m^*}\phi_{ox}^{3/2}}{3hqE_{ox}}\right) \tag{2.10}$$

where E_{ox} is the field across the oxide, ϕ_{ox} is the barrier height for electrons in the conduction band, and m^* is the effective mass of an electron in the conduction band of silicon. The equation governing the current density of the direct tunneling is given by [70]:

$$I_{DR} = AE_{ox}^2 \exp\{-\frac{B[1-(1-\frac{V_{ox}}{\phi_{ox}})^{3/2}]}{E_{ox}}\} \tag{2.11}$$

where V_{ox} is the voltage across the oxide, $A = q^3/(16\pi^2 h\phi_{ox})$, and $B = (4\sqrt{2m^*}\phi_{ox}^{3/2})/(3hq)$.

2.2.3.4 Injection of Hot Carriers from Substrate to Gate Oxide

In a short-channel transistor, due to a high electric field near the *Si-SiO*$_2$ interface, electrons or holes can gain sufficient energy from the electric field to cross the interface potential barrier and enter into the oxide layer. This effect is known as hot-carrier injection.

2.2.3.5 Gate-Induced Drain Leakage (GIDL)

GIDL is due to a high field effect in the drain junction of an MOS transistor. A thinner oxide thickness and higher V_{DD} (higher potential between gate and drain) enhance the electric field and therefore increase GIDL. GIDL is worse for moderate drain doping (in between the extremes previously mentioned), where both the electric field and depletion width (tunneling volume) are considerable. Very high and abrupt drain doping is preferred for minimizing GIDL, as it provides lower series resistance required for high transistor drive currents.

2.2.3.6 Punchthrough

In short-channel devices, due to the proximity of the drain and the source, the depletion regions at the drain-substrate and source-substrate junctions extend into the channel. An increase in the reverse bias across the junctions also pushes the junctions nearer to each other. When the combination of channel length and reverse bias leads to the merging of the depletion regions, punchthrough is said to have occurred.

The device parameter commonly used to characterize the punchthrough is the punchthrough voltage V_{PT}, which estimates the value of V_{DS} for which the punchthrough occurs (i.e., the subthreshold current reaches a particular value) at $V_{GS} = 0$. It is roughly estimated as the value of the V_{DS} for which the sum of the widths of the drain and source depletion regions is equal to

the effective channel length [69]:

$$V_{PT} \propto N_B(L - W_j)^3 \tag{2.12}$$

where N_B is the doping concentration in the bulk; L is the channel length; and W_j is the junction width.

2.3 Technology Trends of Power Dissipation in CMOS Circuits

CMOS technology has to keep scaling down to improve the circuit performance and reduce the cost. As technology scales downward, the transistor density and circuit frequency all increase dramatically. So, the supply voltage V_{DD} must also scale down to reduce dynamic power and maintain reliability. However, this requires the scaling of V_{th} to maintain a reasonable gate overdrive. The scaling of transistor size, V_{DD}, and V_{th} all have a big effect on both the dynamic and leakage power of CMOS circuits. Not only their absolute values, but also their relative magnitudes change dramatically, which has a big impact on CMOS circuits design.

2.3.1 Dynamic Power vs. Leakage Power

As technology scales below 90*nm*, transistor density will continue to double, allowing higher integration. Transistor delay will also continue to improve, at least modestly to 30% reduction per generation. Supply voltage (V_{DD}) will continue to scale modestly by 15%, not by the historic 30% per generation, due to the difficulties in scaling threshold voltage V_{th} and to meet transistor performance goals. Figure 2.3 shows growth in active power of a microprocessor assuming historical $2\times$ growth in number of transistors and with hypothetical $1.5\times$ growth [35].

Subthreshold leakage increase exponentially with the reduction of V_{th}. Assume that V_{th} decreases by 15% per generation, the subthreshold leakage current I_{off} will increase by 5 times each generation. Figure 2.4 projects the *source-drain* (SD) subthreshold leakage power of the microprocessor with $2\times$ and $1.5\times$ transistor growth. Except for the skyrocketing subthreshold leakage, gate leakage becomes larger than $100A/cm^2$ as the physical gate oxide thickness approaches sub-10$\dot{A}$ regime. Junction leakage is also increasing dramatically as channel dopping concentrations approach $5\times10^{18}cm^{-3}$ in the channel [71]. Overall, leakage power increases exponentially with technology scaling.

Since dynamic power remains constant and leakage power increases exponentially with technology scaling, leakage power is becoming dominant in sub-90*nm* CMOS technologies.

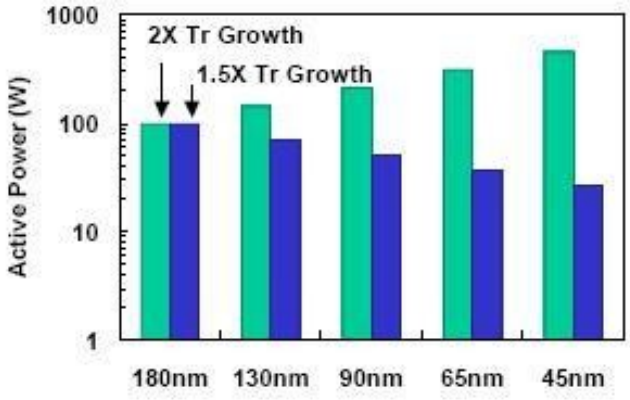


Figure 2.3: Dynamic Power Trend [35]

.

This poses serious challenges for deep submicron CMOS VLSI circuit design. Leakage reduction techniques have to be applied to put the leakage power under control.

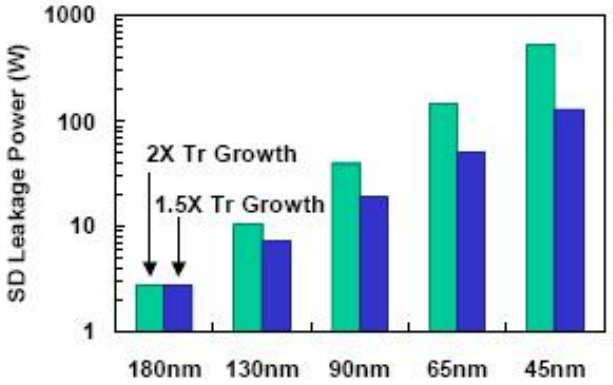


Figure 2.4: Leakage Power Trend [35]

.

2.3.2 Relative Magnitudes of Different Leakage Power Components

The three major types of leakage mechanisms are: subthreshold leakage, gate leakage, and *pn* junction reverse-bias band-to-band tunneling (BTBT) leakage [7]. Although they all increase rapidly with technology scaling, their relative magnitudes will change dramatically. Figure

2.5 shows the contribution of different leakage components in *n*MOS devices at different technology generations [7]. We see that subthreshold leakage is dominant in 90*nm* technology. However, gate leakage becomes equally important in 50*nm* technology and BTBT leakage is also very significant. As the technology scales down to 25*nm*, all three components become nearly equally important. So, each leakage reduction technique needs reevaluation in scaled technologies as the relative magnitudes of different leakage components change.

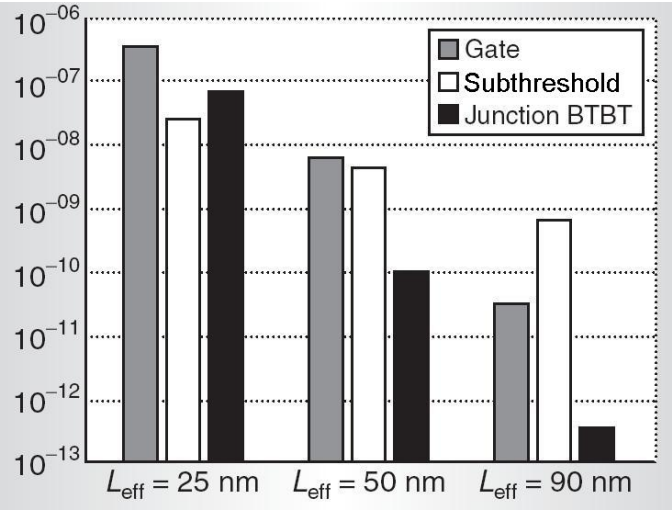


Figure 2.5: Contribution of Different Leakage Components in *n*MOS Devices at Different Technology Generations [7]

.

2.4 Leakage Power Reduction Techniques

The reduction in leakage current has to be achieved using both process and circuit-level techniques. At the process level, leakage reduction can be achieved by controlling the dimensions (length, oxide thickness, junction depth, etc.) and doping profiles in transistors. At the circuit level, threshold voltage and leakage current of transistors can be effectively controlled by controlling the voltages of different device terminals [drain, source, gate, and body (substrate)].

2.4.1 Device-Level Leakage Reduction Techniques

Well engineering is always used to improve short-channel characteristics. By changing the doping profile in the channel region, the distribution of the electric field and potential contours

can be changed. The goal is to optimize the channel profiles to minimize the OFF-state leakage while maximizing the linear and saturated drive currents. Supersteep retrograde wells and halo implants have been used as a means to scale the channel length and increase the transistor drive current without causing an increase in the OFF-state leakage current [32, 77, 82, 91].

2.4.1.1 Retrograde Doping

Retrograde channel doping is a vertically nonuniform, low-high channel doping. It is used to improve the *short channel effects* (SCEs) and to increase surface channel mobility by creating a low surface channel concentration followed by a highly doped subsurface region. The low surface concentration increases surface channel mobility by minimizing channel impurity scattering while the highly doped subsurface region acts as a barrier against punchthrough.

Figure 2.6 shows a schematic band-bending diagram at the threshold condition of an extreme retrograde profile with an undoped surface layer of thickness. For the same gate depletion width, the surface electric field and the total depletion charge of an extreme retrograde channel is one-half that of a uniformly doped channel. This reduces the threshold voltage and improves mobility.

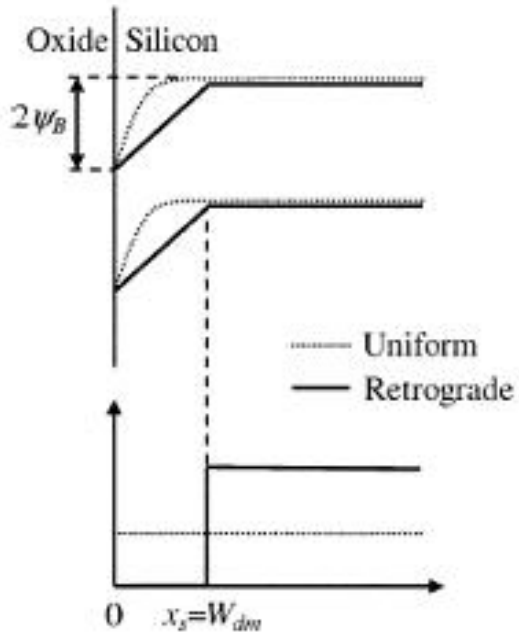


Figure 2.6: Band Diagrams (Shown on Top) at the Threshold Condition for a Uniformly Doped and an Extreme Retrograde-Doped Channel (Doping Profiles Shown at Bottom) [75]

2.4.1.2 Halo Doping

Halo doping or nonuniform channel profile in a lateral direction was introduced below the 0.25μm technology node to provide another way to control the dependence of threshold voltage on channel length. For *n*-channel MOSFETs, more highly *p*-type doped regions are introduced near the two ends of the channel as shown in Figure 2.7.

Under the edges of the gate, in the vicinity of what will eventually become the end of the channel, point defects are injected during sidewall oxidation. These point defects gather doping impurities from the substrate, thereby increasing the doping concentration near the source and drain ends of the channel [28]. A more highly doped *p*-type substrate near the edges of the channel reduces the charge-sharing effects from the source and drain fields, thus reducing the width of the depletion region in the drain-substrate and source-substrate regions. Reduction of charge-sharing effects reduces the threshold voltage degradation due to channel length reduction. Thus, threshold voltage dependence on channel length becomes more flat and the off-current becomes less sensitive to channel length variation. The reduction in drain and source junction depletion region widths also reduces the barrier lowering in the channel, thus reducing DIBL. Since the channel edges are more heavily doped and junction depletion widths are smaller, the distance between source and drain depletion regions is larger. This reduces the punchthrough possibility [75].

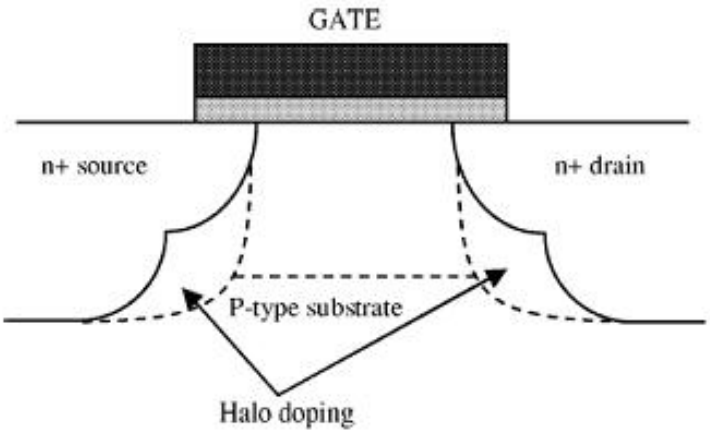


Figure 2.7: Halo or Nonuniform Channel Doping

2.4.2 Circuit-Level Leakage Reduction Techniques

In this section, we will review six major circuit design techniques for leakage reduction in digital circuits: transistor stacking, input vector control, multiple V_{th}, supply voltage scaling (multiple and dynamic V_{DD}), power cut-off, and dynamic power-gating using the Shannon Expansion.

2.4.2.1 Transistor Stacking

Subthershold leakage current flowing through a stack of series-connected transistors reduces when more than one transistor in the stack is turned off. This effect is known as the stacking effect, which is shown in Figure 2.8. The technique of inserting an extra series connected transistor in the pulldown path of a gate and turning it off in the standby-mode of operation is known as forced stacking [33]. The extra transistor is turned on during the regular mode of operation and turned off during the idle mode of operation. When the extra transistor is turned off, the intermediate source voltage increases, which results in a decrease in the subthreshold current through the top transistor. Hence, the total subthreshold leakage through a two-transistor stack is reduced. Forced stacking only works for standby leakage power reduction.

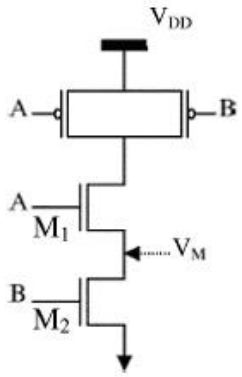


Figure 2.8: Stacking Effect in Two-Input NAND Gate

Another way of using the stacking effect for leakage reduction is to replace a single transistor with two transistors of the same size. This is equivalent to replacing a low threshold transistor with a high threshold transistor in the dual-threshold transistor technique. Static timing analysis is needed to identify those gates on non-critical paths for possible insertion of stacking

transistors. Similar algorithms as high threshold transistor insertion in the dual-threshold transistor technique can be used. Please refer to Section 2.4.2.3 for the detail introduction of the dual-threshold transistor technique.

2.4.2.2 Leakage Reduction by Input Vector Control

Due to the stacking effect, the subthreshold leakage through a logic gate depends on the applied input vector. This makes the total leakage current of a circuit dependent on the states of the primary inputs [25]. It has been shown that the leakage current ratio between different input combinations can be as high as 10. The goal can then be expressed as finding the input pattern that maximizes the number of disabled (off) transistors in all stacks across the circuit [90].

One possible way is to perform an exhaustive circuit-level simulation for all input patterns to find the pattern with the minimum leakage current. However, this approach is not practical for large circuits. Z. Chen *et al.* proposed a genetic algorithm [21] to locate the vector that results in the near minimal leakage current. J. Halter and F. Najm [29] used probabilistic methods to reduce the number of simulations necessary to find a solution with a desired accuracy. SAT-based formulation [3, 4, 8] were also proposed for finding the minimum leakage vector at the circuit inputs.

2.4.2.3 Leakage Reduction by Multiple Threshold Voltage Designs

One way of decreasing the leakage current is to increase the threshold voltages of transistors. Multiple-threshold CMOS technologies, which provide both high- and low-threshold transistors in a single chip, can be used to deal with the leakage problem. The high-threshold transistors can suppress the subthreshold leakage current, while the low-threshold transistors are used to achieve high performance. Several multiple-threshold circuit design techniques have been developed recently, including multi-threshold CMOS, dual-threshold CMOS, variable threshold CMOS, and dynamic threshold CMOS.

Multi-Threshold Voltage CMOS.

Multi-threshold voltage CMOS (MTCMOS) reduces the leakage by inserting high-threshold devices in series with low-threshold circuitry [54]. Figure 2.9 shows the schematic of an MTCMOS circuit. In the active mode, the sleep control transistors (MP and MN) are turned on.

Since their on-resistances are small, the virtual supply voltages (*VDDV* and *VSSV*) almost function as real power lines. In the standby mode, MN and MP are turned off, and the leakage current is low.

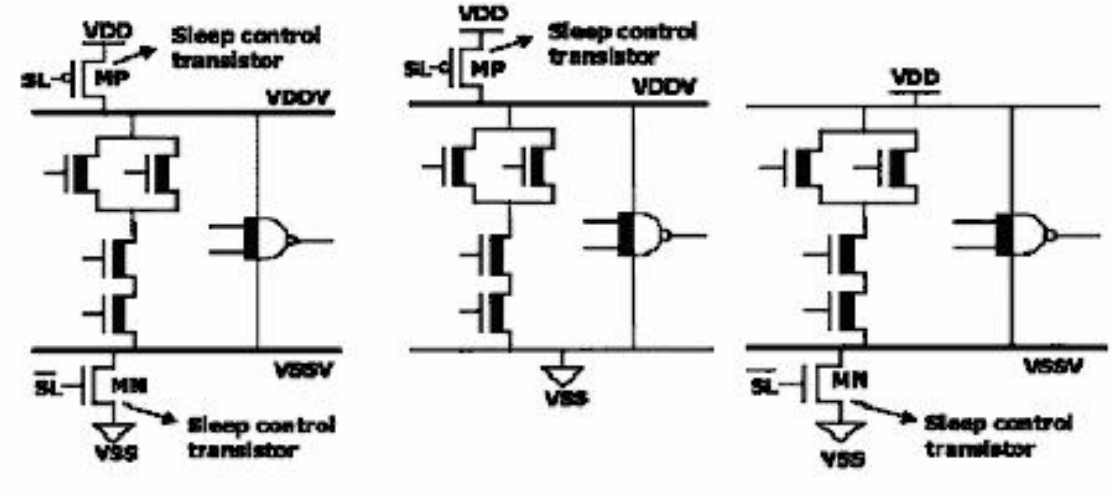


Figure 2.9: (a) Original MTCMOS (b) *p*MOS Insertion MTCMOS (c) *n*MOS Insertion MTCMOS [54]

In fact, only one type of high transistor is enough for leakage control. Figures 2.9 (b) and (c) show the *p*MOS insertion and *n*MOS insertion schemes, respectively. The *n*MOS insertion scheme is preferable, since the *n*MOS on-resistance is smaller at the same width; therefore, it can be sized smaller than the corresponding *p*MOS. This technique is only effective for standby leakage power reduction.

Dual-Threshold CMOS.

Another approach of MTCMOS is to use high-threshold voltage devices on noncritical paths to reduce the leakage power while using low-threshold devices on critical paths so that the circuit performance is maintained. This technique has been called dual-threshold CMOS [20]. It is an integer linear program to choose an optimal assignment of dual-V_{th} for all of the transistors or gates in the circuit. Various heuristic algorithms are proposed to solve this problem for big circuits [51, 74, 84, 86, 87]. Dual-threshold CMOS is a very effective approach for leakage reduction in both active mode and standby mode. More than 80% of leakage power savings have been reported. Compared with other leakage reduction techniques, it requires very little

modification of the circuit design. It can also be combined with transistor sizing and multiple V_{DD} to get more leakage power savings [31, 36, 39, 56, 59, 60, 72, 72, 88]. Y. Lu *et al.* combine dual-V_{th} assignment with path balancing using integer linear programing to reduce both leakage and dynamic glitch power simultaneously [47–50]. Thus, dual-threshold CMOS is widely used in modern CMOS fabrication lines.

Variable Threshold CMOS.

Variable threshold CMOS (VTMOS) is a technique, which uses the body bias voltage to change the threshold of CMOS transistors [43]. It has been reported that reverse body biasing lowers integrated circuit leakage by three orders of magnitude in a 0.35μm technology [38]. However, it was also shown that the effectiveness of reverse body bias in lowering leakage decreases as technology scales. This technology also requires routing the body grid, which will add to the overall chip area.

Dynamic Threshold CMOS.

In *dynamic threshold CMOS* (DTMOS), the threshold voltage is altered dynamically to suit the operating state of the circuit. It can be achieved by tying the gate and body together [9]. DTMOS can be developed in bulk technologies by using triple wells. Doping engineering is needed to reduce the parasitic components [85]. The supply voltage of DTMOS is limited by the diode built-in potential in bulk silicon technology. The *pn* diode between source and body should be reverse biased. Hence, this technique is only suitable for ultra-low voltage (0.6V and below) circuits in bulk CMOS. Another way for dynamic threshold design is to control the body bias voltage dynamically through a bias-control circuit depending on the workload of the system. When the workload becomes less, the bias control circuit will change the body bias to increase the threshold to reduce the power [40].

2.4.2.4 Leakage Reduction by Power Cut-Off

Instead of using low V_{DD} for active mode and high V_{DD} for standby mode, the power supply can be cut-off during the standby state and resumed during the active mode. This is called power cut-off technology. Two different power cut-off CMOS technologies have been proposed: *super cut-off CMOS* (SCCMOS) [37] and *zigzag super cut-off CMOS* (ZSCCMOS) [53].

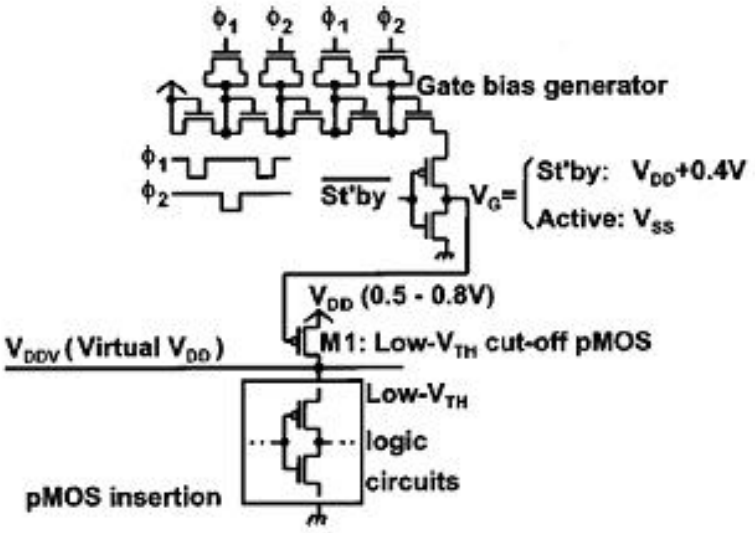


Figure 2.10: Concept of SCCMOS

The SCCMOS scheme was proposed and demonstrated to achieve high speed and low standby current with sub-1V supply voltages. In Figure 2.10, the low-V_{th} cut-off *p*MOS, M1, whose V_{th} is 0.1-0.2V, is inserted in series to the logic circuits consisting of low-V_{th} MOSFETs. The gate voltage of M1, V_G, is grounded in an active mode to turn M1 on. When the logic circuits enter standby operation, V_G is overdriven to V_{DD}+0.4V to completely cut off the leakage current.

A problem associated with this scheme is that data can get lost during the long sleep period due to the leakage current. SCCMOS also suffers from a long wake-up time and a high current peak at the sleep-to-active transition. This is due to the virtual V_{DD} node being discharged (charged) during the sleep period and being charged (discharged) when returning to active mode. A *zigzag super cut-off CMOS* (ZSCCMOS) method was then proposed to improve the operating speed by eliminating the series-connected switches while achieving the relaxation of the high-voltage stress at the cut-off switch [53].

Tschanz *et al.* incorporated the power cut-off technology with the clock-gating scheme for leakage power reduction in a microprocessor [78]. The gated-clock signal is used to synchronize the power cut-off controls of the respective circuit blocks, so that not only dynamic power but also leakage power can be reduced when the circuit block is in standby mode.

2.4.2.5 Dynamic Power Gating Using the Shannon Expansion

Bhunia *et al.* proposed an active leakage reduction technique using supply gating [12]. They use the Shannon expansion to identify the idle part of the circuit and dynamically apply supply gating to those idle parts so that active leakage power is saved. Based on the Shannon expansion, each function $f(x_1, x_2, ..., x_n)$ can be expanded into two parts based on variable x_i:

$$\begin{aligned} f(x_1, x_2, ..., x_n) &= x_i CF_1 + x_i' CF_2 \\ CF_1 &= f(x_1, x_2, ..., x_i = 1, ..., x_n); \\ CF_2 &= f(x_1, x_2, ..., x_i = 0, ..., x_n); \end{aligned} \tag{2.13}$$

Based on the above expansion, $f(x_1, x_2, ..., x_n)$ can be implemented in a circuit shown in Figure 2.11. For such an implementation, x_i acts as a power gating signal for the circuit and only half the circuit is active at any time. In a big combinational circuit, all of the Boolean functions could be expanded by applying the Shannon expansion recursively and implemented into a similar circuit architecture. Thus, only a partial circuit will be active at any time and active leakage power is saved. With this technique, 15% to 88% total power reduction in MCNC benchmarks are reported.

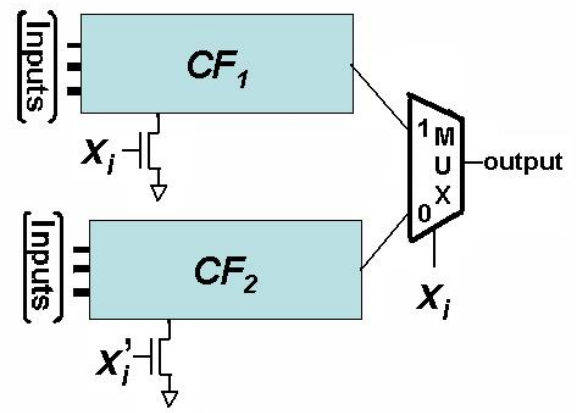


Figure 2.11: Dynamic Supply Gating Using the Shannon Expansion

2.4.3 Summary

All of the techniques described above can be used to reduce the leakage power of a circuit in standby mode. However, even when the circuit is active, it still consumes a significant

amount of leakage in deep-submicron CMOS technologies. In fact, the leakage power in the active mode is significantly larger due to higher die temperature in active mode. Among the technologies we described above, dual-threshold CMOS, DTMOS, and all of the device-level techniques are effective for active leakage reduction. Most of the other schemes only work for standby leakage reduction. However, the dual-threshold technique does not reduce the leakage on critical paths. Thus, it does not help much for timing-optimized circuits, whose paths are usually well balanced. The DTMOS is usually achieved by tying the gate and body together [9]. So, the supply voltage of DTMOS is limited by the diode built-in potential in bulk silicon technology. Hence, this technique is only suitable for ultra-low voltage (0.6V and below) circuits in bulk CMOS. Thus, more effective active leakage reduction techniques are still very desirable. Here we are proposing a new leakage reduction technology, called the *dynamic power cutoff technique*, which reduces both active leakage and standby leakage.

Chapter 3

Prior Work: Power Grid and Process Variation Analysis

3.1 Power Grid Analysis

As the supply voltage and threshold voltage are decreasing with technology scaling, checking the integrity of the voltage on the power distribution network is becoming crucial. With lower supply voltages, smaller voltage drops become more significant and can cause longer delays and lead to soft errors. Voltage drop on the power distribution network is mainly due to *IR* drop and Ldi/dt drop. The *IR* drop is due to the resistance of the metal lines of the power network. The Ldi/dt drop is due to the self and mutual inductances of the power lines.

With technology scaling, the wire resistances of proportionately scaled wires have increased significantly. The inductive behavior seen in global lines is also getting severer with the rapid increase of the circuits' operating frequency. Meanwhile, the current density and the total current increase due to smaller devices and larger dies. And, the higher switching speed of smaller transistors produces faster current transients in the power distribution network. The high currents cause large *IR* voltage drops while the fast current transients cause large inductive voltage drops (Ldi/dt noise) in the power distribution network. These, altogether, make it more and more challenging to maintain a highly stable power supply voltage. Typically, the overall noise on the power distribution network has to be less than 5% or 10% of the supply voltage.

Power distribution networks in high-performance digital ICs are commonly structured as a multilayer grid, called the *power grid* (PG). In such a grid, straight power/ground lines in each metalization layer span the entire die (or a large functional unit) and are orthogonal to the lines in the adjacent layers. The power and ground lines typically alternate in each layer. Vias are used to connect a power (ground) line to another power (ground) line at the overlap sites.

3.1.1 Power Grid Modeling

The power grid is usually modeled as a *RLC* network shown in Figure 3.1, where each

branch of the power grid is represented by a resister R_{pg}, an inductor L_{pg}, and a capacitor C_{pg} [89]. In addition, some grid nodes have ideal voltage sources (to ground) representing the connection to the external voltage supply and some grid nodes have ideal current sources (to ground) representing the currents drawn by the circuits tied to the grid at those nodes.

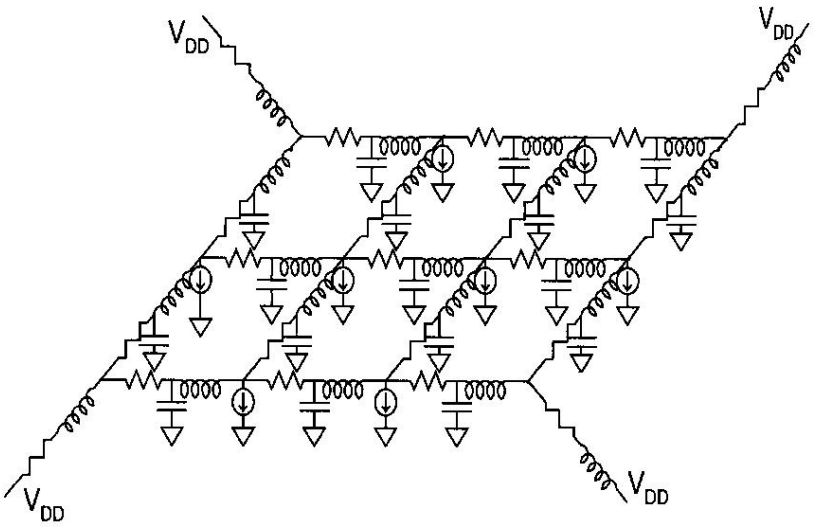


Figure 3.1: *RLC* Model of Power Grid

3.1.2 Prior Power Grid Analysis Techniques

The simulation of the power grid network requires solving a large system of differential equations that can be reduced to a linear algebra system using a Taylor expansion. As the current-day supply networks may contain millions of nodes, solving such a huge linear system is very challenging. Traditional analog simulators, such as *SPICE*/*HSPICE*/*SPECTRE*, can only be used to simulate very small power grid networks. Many algorithms have been proposed to solve those large power grid networks more efficiently.

Since the computational complexity of direct methods to solve linear systems of size n is $O(n^3)$, sparsity and the grid structure in the power distribution network are usually exploited to reduce computational complexity. A preconditioned conjugate gradient iterative method, using incomplete Cholesky factorization as the pre-conditioner, was described by Chen and Chen [19]. Although this pre-condition-based iterative method reduces the computational complexity of DC analysis of power grids from $O(n^3)$ to $O(n^2)$, it is not efficient for transient analysis since it is not possible to leverage previous simulation runs.

A multi-grid approach described by Kozhaya *et al.* [42] exploits the grid structure by mapping the original system to a coarsened grid, solving the coarsened grid, and remapping it back to the original grid. The solution of the original system through remapping is obtained through an interpolation procedure. However, in the absence of error bounds, this method may not always be accurate. Moreover, the effort to keep track of the geometrical information of the power grid is expensive, further limiting its applications.

Algebraic multigrid methods were proposed in [73, 92] to handle general network topologies. Algebraic multi-grid methods can be thought of as iterative solvers that use the multi-grid operator as a pre-conditioner. In such methods, the computational cost in each time step of the transient analysis is comparable to that for DC analysis, making it unsuitable for efficient transient analysis.

Other approaches to power grid analysis include those based on random walks, model order reduction, and hierarchical analysis. Statistical techniques based on random walks [44, 62] are very fast but suffer from accuracy loss and convergence issues. Model order reduction methods are inefficient for power grid simulation due to (i) a large number of external terminals and (ii) the loss of sparsity in the reduced model [27]. Hierarchical techniques are applicable if the power grid is not flattened, and macro-models for local grids can be built to speed up simulation at the global level [16, 92].

3.2 Process Variation Analysis

Process variations are posing an increasing challenge to the design, analysis, and testing of nano-scale VLSI circuits. This is mainly due to the ever-increasing variabilities in the process parameters, such as channel length, transistor width, oxide thickness, and the random placement of dopants in the channel. In general, process variations can be classified into two main categories: inter- and intra-die variations.

With inter-die variations, the same device on a die can have different characteristics across different dies. Intra-die variations, on the other hand, are the variations of transistor characteristics within a single die. Traditionally, inter-die variations have been the main concern in CMOS digital circuit design, and intra-die variations have been neglected [26]. However, with CMOS technology scaling down to sub-100*nm* features, intra-die variations are becoming much more significant than the inter-die variations [14].

Intra-die variation can he further divided into two catergories: random variations, and spatially correlated variations. Random intra-die variations have no dependence on the location of the devices, while intra-die variations that are spatially correlated produce an increased likelihood of similar values for devices that are closely spaced versus those that are placed further apart [6].

3.2.1 Process Variation Modeling

To model process variations, the transistor length L and width W, gate oxide thickness t_{ox}, and threshold voltage V_{th} are usually modeled as normal distributed random variables [55]. Truncated normal distributions are usually used for the above parameters to reflect the fact that the process variations in an operational chip cannot be more than some finite maximum value. People also assume that the variations of above parameters are mutually independent.

$$L_{total,k} = L_{nom} + \Delta L_{inter} + \Delta L_{intra,k} \tag{3.1}$$

Equation 3.1 shows an example for modeling the length L of device k under process variations. In this equation, $L_{total,k}$ is the length of device k, L_{nom} is the mean of the length of all devices across all possible dies, ΔL_{inter} is the inter-die device length variation, and $\Delta L_{intra,k}$ is the intra-die length variation for device k. Note that ΔL_{inter} is the same for all devices on a die and $\Delta L_{intra,k}$ is different for different devices on a die. However, $\Delta L_{intra,k}$ has the same distribution for all devices on a die.

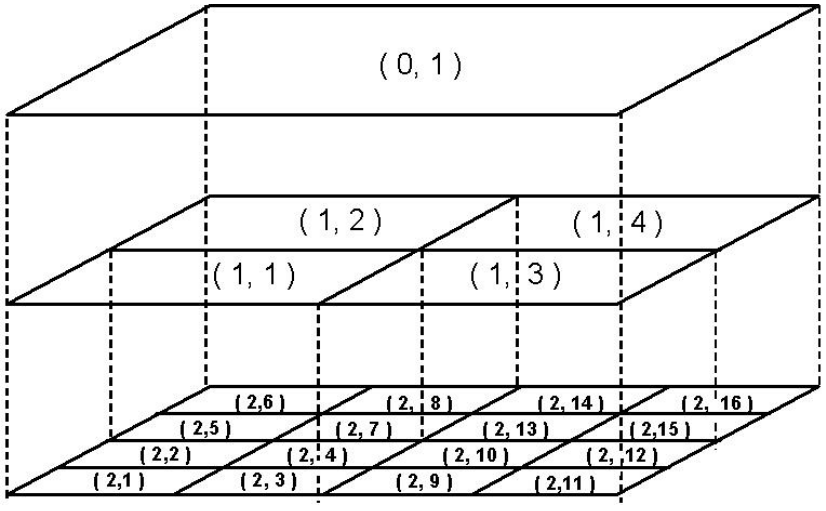


Figure 3.2: Modeling Spatial Correlations Using Quad-Tree Partitioning [6]

To model the spatial correlations of intra-die process variations, Agarawal *et al.* proposed the multi-level grid model [6]. In this model, the area of the die is divided into regions using a multi-level quad-tree partitioning, as shown in Figure 3.2. For each level l, the die area is partitioned into 2^l *by* 2^l squares, where the top level 0 has a single region and the bottom level m has 4^m regions. The process variation of a transistor in any grid at the bottom level is then composed as the sum of the variation in that particular grid and the variations in all of its parent grids. For example, the variation of the channel length of transistors in grid $(2,6)$ is represented as:

$$\Delta W_{variation}(2,6) = \Delta W_{2,6} + \Delta W_{1,2} + \Delta W_{0,1} \tag{3.2}$$

where $\Delta W_{variation}(2,6)$ is the total variation in the width of transistors in grid $(2,6)$; $\Delta W_{2,6}$, $\Delta W_{1,2}$, and $\Delta W_{0,1}$ represent the variation in the width in grids $(2,6)$, $(1,2)$, and $(0,1)$, respectively.

Using this multi-level grid model, transistors that lie within closer proximity of each other will have more common intra-die variation components resulting stronger intra-die correlations. Also, the variations in grid $(0,1)$ model the inter-die variations since it is the parent grid of all other grids. So, both inter- and intra-die variations are represented using this model.

3.2.2 Statistical Static Timing Analysis

Static timing analysis (STA) is widely used in timing analysis of VLSI circuits. Traditionally, discrepancies in VLSI chip parameters have been accounted for in STA using corner analysis, such as best-case, nominal, and worst-case, and so on. However, the corner analysis method is becoming unacceptably conservative and overly-pessimistic due to the ever-increasing process variations. It was shown that worst-case analysis overestimates path delays by more than 50% [52]. So, *statistical static timing analysis* (SSTA) has been proposed to replace the traditional STA to give more accurate prediction of the timing performance of the circuits.

To do SSTA for a circuit, the underlying process parameters, such as channel length, width, oxide thickness, and the threshold voltage, are all modeled as *random variables* (RVs). Then, the delays of gates, paths, and circuits are all RVs as well. Given certain distributions of the process parameters, we can get a certain delay distribution of each gate, path, and circuit, which gives us a better prediction of the circuit performance and a better metric for circuit design.

3.2.3 Prior Statistical Static Timing Analysis Techniques

The most straightforward way to do SSTA is the Monte Carlo method. By generating a large number of samples of the process parameters, each with a certain distribution, and doing STA for each sample of parameters, we can get a set of samples of the gate, path, and circuit delays, which show us the distribution of those delays. Although, Monte Carlo based SSTA is very accurate as long as enough samples are analyzed, it is prohibitively slow for large circuits. So, many more efficient SSTA algorithms have been proposed. Most of them involve propagating distributions of delay through the logic network. Based on different ways of propagating the delay distributions, these SSTA approaches can be further classified into two categories: path-based SSTA and block-based SSTA.

Path-based SSTA [6, 45, 57, 58] seeks to estimate timing statistically on selected critical paths. However, the task of selecting a subset of paths whose time constraints are statistically critical has a worst case computational complexity that grows exponentially with respect to circuit size. Hence, path-based SSTA is not easily scalable to handle realistic circuits.

Block-based SSTA [5, 6, 18, 24, 83], on the other hand, champions the notion of progressive computation. Specifically, by treating every gate/wire as a timing block, SSTA is performed block by block in the forward direction in the circuit timing graph without looking back to the path history. As such, the computational complexity of block-based SSTA would grow linearly with respect to circuit size.

Chapter 4

Novel *Dynamic Power Cutoff Technique* (DPCT)

4.1 Introduction

Here, we propose a novel active leakage power reduction technique called DPCT based on power cutoff [10]. We first identify when a gate is idle by finding its switching window using static timing analysis, and then we turn off the power of each gate when it is idle within each clock cycle. In this chapter, we first introduce the basic idea of DPCT, then we discuss its implementation, and finally we discuss its power savings.

4.2 Basic Idea of DPCT

We observed that each logic gate only switches within a particular timing window during each clock cycle even when the circuit is in active mode. We call this the *switching window* of a gate. If we turn on the power connection of each gate only during its switching window during each clock cycle, we can save part of the active leakage power with very little effect on its normal transitions, usually a little extra delay in 70*nm* CMOS technology. The potential of active leakage power saving in a CMOS gate by doing this is proportional to the ratio of the power off time of the gate to the clock period. The possible power off time for a gate is equal to the clock period minus the switching window of the gate. This is the basic idea of DPCT. Figure 4.1 shows the basic architecture of a circuit with DPCT added.

We use both *n*MOSFET and *p*MOSFET low threshold device insertion to increase the leakage savings. If we left out the *GND* cutoff transistor, when a logic gate output is high, the *p*-tree is on and the *n*-tree is off. Therefore, a leakage path exists from the high output through the *n*-tree to *GND*. A similar argument holds for the V_{DD} cutoff transistor when the gate output is low, so we need both cutoff transistors. A circuit is partitioned into different groups based on the switching windows of each gate. Gates with the same switching window are treated as one group and the power connections of all gates within the same group are controlled by one pair

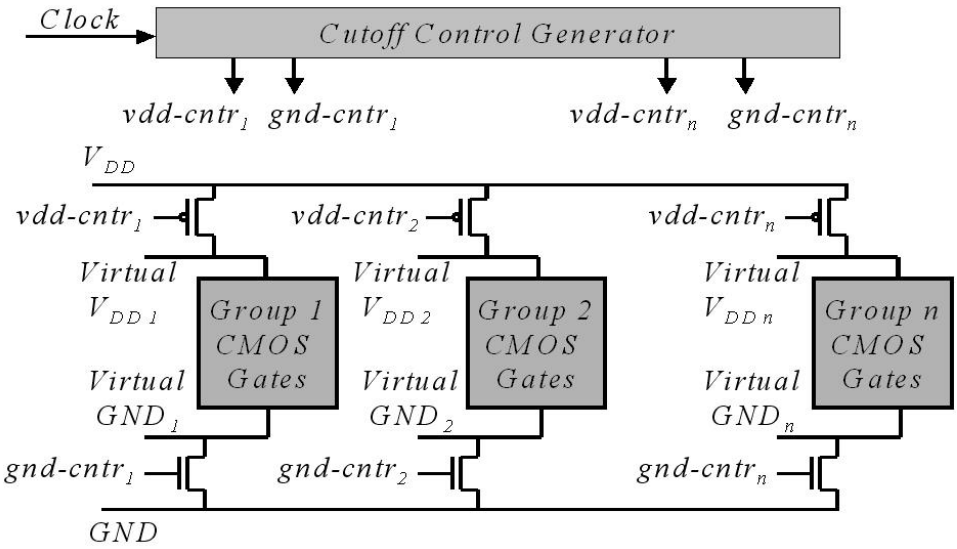


Figure 4.1: Architecture of a Circuit with DPCT

of power cutoff MOSFETs, a *p*MOSFET and an *n*MOSFET. All such different groups make a partition of the circuit. There is one pair of cutoff control signals for each group, *vdd-cntr*$_i$ and *gnd-cntr*$_i$, to control V_{DD} and *GND* of the gates in that group. The cutoff control signals are generated by the *cutoff control generator* using the global clock signal. They all have the same period as the global clock and are carefully tuned so that they turn on the power cutoff MOSFETs only during the switching window of that group within each clock cycle. Suppose that the global clock period is $1GHz$ with a 50% duty cycle. The waveforms in Figure 4.2 show the relationship of the global clock and one pair of cutoff control signals, which control a group whose switching window is $(60ps, 180ps)$.

4.3 Six Steps to Implement DPCT

There are several problems in directly applying the above basic idea to implement DPCT. First, the widths of switching windows of many gates are almost as big as the clock period. So, the possible power off time of many gates is almost 0, which gives little leakage power savings by applying DPCT. Second, there may be hundreds, even thousands, of switching windows within each circuit. It will be very clumsy and expensive to add so many cutoff control signals and cutoff control MOSFETs in a circuit. So, we propose a six-step approach to implement DPCT, which gives near-optimal leakage power saving with minimal extra cost.

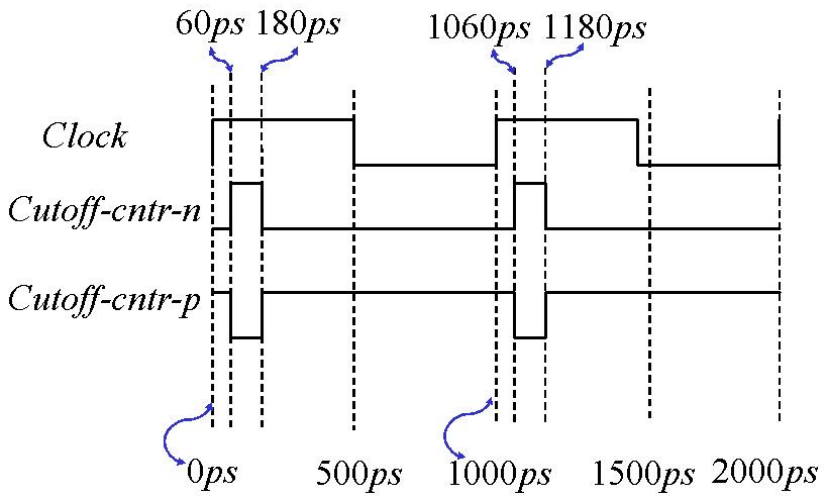


Figure 4.2: The Clock and One Pair of Cutoff Control Signals

4.3.1 1st Step: Calculate the *Minimal Switching Window* of Each Gate by Static Timing Analysis

We first define the *switching window* of a gate based on the timing window method proposed by Raja *et al.* [65]. The timing window method was used successfully for dynamic glitch power reduction in CMOS circuits by path balancing [30, 63, 64, 66–68, 79, 80]. Then we define the *minimal switching window* of each gate.

4.3.1.1 Switching Window Based on Traditional Timing Window

The timing window (t, T) for each circuit node is specified by two variables t and T. Here, t is the earliest time and T is the most delayed time of signal transition at the node. Consider a CMOS gate with n inputs and maximal delay D and minimal delay d in Figure 4.3. Each input has a timing window (t_i, T_i), and the output has a timing window (t_o, T_o). Then, the output node timing window is derived from the timing windows of the inputs and the gate delay:

$$T_o = max(T_i + D), \qquad t_o = min(t_i + d) \tag{4.1}$$

Using Equation 4.1, we calculate the timing windows of all circuit nodes by a level-order traversal from *primary inputs* (PIs) to *primary outputs* (POs), if we know the delay of each gate and the timing window of each PI. The maximum T_o of all POs is the worst-case delay of the circuit. In a real circuit, the clock cycle is determined by the worst-case circuit delay. Usually, a 10% to 15% margin is added to make sure that the circuit can always finish its transitions

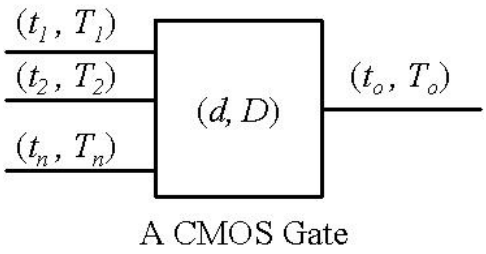


Figure 4.3: Timing Window of a CMOS Gate

even under the worst case.

Based on the timing window method discussed above, we define the *switching window* of a gate as $(min(t_i), T_o)$, where $min(t_i)$ is the earliest arrival time among all inputs, and T_o is the latest arrival of the signal at the output of the gate. The switching window of a gate defines a timing window from the earliest arrival time of its inputs to the end time of the latest possible transition the gate can make. A logic gate is in active mode only within its switching window during each clock cycle.

If we turn on the power of each gate only within its switching window during each clock cycle, we can save part of active leakage power without affecting its normal transition activity except for a little added delay. The percentage of active leakage power saving of a CMOS gate, PS_{gate}, is given by:

$$PS_{gate} = a \times t_{off}/T_{cycle} \tag{4.2}$$

where T_{cycle} is the period of the clock cycle, t_{off} is the power-off time of that gate within each clock cycle, and $0 < a < 1$ is related to t_{off}/T_{cycle}. The bigger t_{off}/T_{cycle}, the closer a is to 1; the smaller t_{off}/T_{cycle}, the closer a is to 0. In our experiments, we calculate a by curve fitting based on the power saving results from analog simulation. From our experiments, we found noticeable leakage savings only when $t_{off}/T_{cycle} > 1/3$. This is because the virtual V_{DD} and *GND* take a little extra time to collapse after the cutoff transistors are turned off. Also, it takes some extra cost to operate the cutoff transistors.

However, the switching window of a gate will become much wider if the gate has very unbalanced minimal and maximal delays, or if its inputs come from different paths with big delay differences, or if some inputs already have wide switching windows. The wide switching windows of the gates will make the switching windows of their fanout gates even wider. The result is that the widths of many gates' switching windows are almost as big as the worst-case

delay of the circuit. If we turn on the power of each gate within its switching window like this in each clock cycle, we cannot save much leakage power.

4.3.1.2 Minimal Switching Window

To solve the problem of the switching window, we propose another type of timing window, named the *minimal switching window* (MSW) of a CMOS gate, which is defined as the minimal timing window during which we can keep the gate on without affecting the logic function and worst-case circuit delay. It is represented by $((T_o - D), T_o)$, where T_o is the latest arrival of the signal at the output of the gate and D is the maximal delay of the gate. The idea is that we do not have to turn on the gate as early as the earliest input signal comes. Actually, the signals that arrive early can wait until the gate turns on. As long as we turn on the gate D time units earlier than T_o, we can guarantee that the transition of its output happens no later than T_o. Because the worst-case delay of the circuit only depends on the latest transition time of each gate, not the earliest transition time, turning on the power of each gate only within its MSW during each clock cycle will not affect the function and the timing performance of the circuit. Of course, cutoff transistors will introduce some extra delay. But this extra delay will always exist regardless of which timing window we use.

The advantage of the MSW is that its width only depends on the maximal delay of the gate itself, which is usually less that $1/10$ of the worst-case circuit delay in big circuits. It does not blow up with the unbalanced delay of the gate and the delay differences of its inputs. By turning on each gate only within its MSW, we can save a large percentage of the active leakage power of the circuit. Furthermore, as we only turn on the gate after all input signals are stabilized, the glitches caused by different input path delays are avoided. This leads to a 9.7% dynamic power savings (see Table 6.2).

To calculate the MSW of each gate, we first calculate each gate delay. Then we use static timing analysis to calculate T_o of each gate. Finally, we apply $((T_o - D), T_o)$ to get the MSW of each gate. We use 70*nm* CMOS Berkeley Predictive Models, a BSIM3v3 model, for our simulation. We model each CMOS gate as an *RC* network, which is the same approach as used by Wei *et al*. [86]. The load capacitance *C* is calculated using the parameters and equations defined in the BSIM3v3 model manual. A look-up table based on SPECTRE^{TM} analog simulation is used to get the equivalent *R* of the *n*-tree or *p*-tree of a CMOS gate based on the gate type, the number of fanins, the number of fanouts and the transistor sizes to compute the equivalent

on-resistance. The delay calculation results from static timing analysis were verified on various benchmark circuits to be within 10% error compared with the results of SPECTRE^{TM} analog simulation. The delay calculation, static timing analysis and MSW calculation are implemented as C programs.

To allow for this 10% delay estimation error and ensure that signals make full swings to logic 1 or 0, we doubled the width of the MSW to be $((T_o - D) - 0.5 \times D, T_o + 0.5 \times D)$. we experimented with timing windows that were 1.0, 1.1, 1.2, 1.3, 1.4, 1.5 and 2.0 times the MSW width. The 1.0, 1.1 and 1.2 figures gave output logic errors because there is not enough overlap between the power-on times of nearby groups. The 1.3 and 1.4 values only work for some of the benchmarks. But, the 1.5 and 2.0 values worked correctly on all benchmarks. To find an appropriate MSW width for a specific circuit, we can start from 1.5 times the MSW and try reducing it to 1.4 or 1.3 until logic errors occur. Since reducing the MSW width from 2.0$\times$ to 1.5$\times$ only increases power savings by 5%, we used 2.0$\times$ to provide a bigger margin for process variations. This also gives 50% overlap of the power-on times between each gate with its fanin gates and fanout gates that are in nearby groups. This allows some early transitions to happen, which can reduce the potential delay cost of DPCT. As the width of the MSW is usually less than $1/10$ of the clock period, doubling the MSW width has little effect on the active leakage power savings.

One special case of using DPCT is that a gate in one group drives another gate in another group, which is several groups away. Figure 4.4 shows such a circuit, where the circuit is partitioned into four groups. Originally, the output of gate 1 drives gate 6. Since there is no overlap between the power-on times of gate 1 and gate 6, signal *A* may collapse before the power of gate 6 is turned on. To solve this problem, an extra *C*-switch has to be inserted to keep the signal steady after gate 1 is turned off, which is shown as gate 7 in this figure. The *C*-switch is wired to the power control signal for *Group* 1, so that when *Group* 1 is off, the *C*-switch is off.

4.3.2 2nd Step: Heuristic Partitioning of the Circuit by Dynamic Programming

Usually, different gates have different MSWs and there may be hundreds, even thousands, of different MSWs within a circuit. The number of MSWs within each *ISCAS '85* benchmark circuits is shown in Table 4.2. Each MSW will need a pair of power cutoff MOSFETs and a pair of power cutoff control signals. Adding so many power cutoff control groups to a circuit

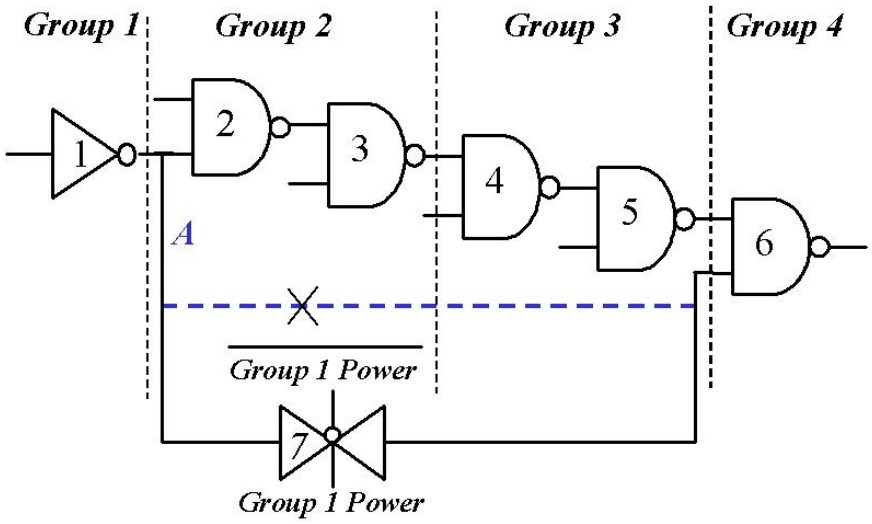


Figure 4.4: A Special Case of DPCT

will be quite expensive because of the extra cost of cutoff MOSFETs and generating the power cutoff control signals. Actually, some groups could be combined to reduce the cost with little effect on leakage power saving.

The switching window of a combined group is the union of the MSWs of all gates within that group. For example, the switching window for a group combining n MSWs (t_i, T_i) will be $(min(t_i), max(T_i)), \quad i = 1, ..., n$. Obviously, combining multiple groups into one group will expand the width of the switching window of some gates in this group, which will result in less leakage power saving according to Equation 4.2. So, the partitioning algorithm has to be able to balance the goal of reducing the extra cost with the purpose of saving more power.

4.3.2.1 The Objective Function to Optimize

To optimize the leakage power saving and the extra cost, we set up a combined objective function given by:

$$OPT = (1 - pb) \times PS - pb \times COST \tag{4.3}$$

where PS is the estimated total active leakage power saving percentage under the current partitioning scheme, $COST$ is the estimated indication of the total area and speed costs under the current partitioning scheme. Here, pb sets the relative weights of PS and $COST$. Adjusting the value of pb allows us to choose whether we want to optimize for more power saving or for less cost.

The *COST* is an indication of the area and delay cost of a partitioning scheme that we use for the heuristic partitioning algorithm. It is proportional to the number of groups and the switching window width of each group. We define *COST* as:

$$COST = \sum_{k=1}^{N_{groups}} pcost \times Width_k / T_{cycle} \tag{4.4}$$

where N_{groups} is the total number of groups under the current partitioning scheme, $Width_k$ is the width of the switching window of group k under the current partitioning scheme, and *pcost* is the overall cost per group per unit time of the switching window. Together *pcost* and *pb* set the relative weight of the *COST*. We set $pcost = 0.1$ and $pb = 0.67$ in our experiments because this gives above 80% average leakage savings with less than 13% *COST* on *ISCAS '85* benchmark circuits. Table 4.1 shows the estimated average leakage power savings *PS* and the *COST* under different *pb* values when $pcost = 0.1$. Note that the *COST* we defined here is influenced by N_{groups} and the switching window widths. It is not the real area cost or delay cost of DPCT, but it is proportional to the real area cost given in Table 6.2.

Table 4.1: Estimated Average Power Savings and Cost vs. *pb* for *ISCAS '85* Circuits

pb	Average Estimated		Average # of	
	Leakage Power Saving	*COST*	Groups	Gates per Group
0.33	84.9%	17.3%	35	41
0.50	82.9%	14.7%	25	58
0.60	82.0%	13.6%	21	70
0.67	81.3%	12.9%	18	81
0.71	79.2%	12.6%	17	87
0.75	78.6%	12.4%	16	95

To simplify the calculation of the leakage power, we assume that each gate consumes equal amounts of leakage. Although this is a rough estimate, it is good enough for our heuristic partitioning algorithm. The total leakage saving of a partitioning scheme is given by:

$$PS = \sum_{k=1}^{N_{gates}} a \times (T_{cycle} - Width_k) / T_{cycle} \tag{4.5}$$

where N_{gates} is the total number of gates within this circuit, $Width_k$ is the width of the switching window of the group where the gate k belongs under the current partitioning scheme, T_{cycle} is the clock period, and a is a parameter for estimation of leakage power saving. We calculate a by comparing the estimated active leakage power savings with the simulation results from

NanoSim^{TM}. Based on our experiments, $a = 0.978$ is a good empirical value to match the analog simulation results. Here a is very close to 1 because the ratio of t_{off}/T_{cycle} in Equation 4.2 is small, usually less than $1/10$ in our cases. To calculate a, we first calculate the average of the estimated active power savings of DPCT on all *ISCAS '85* benchmark circuits using Equation 4.5 with $a = 1$, which is defined as PS_{est}. In our case, we got $PS_{est} = 86.3\%$. We also define the average of the active power savings of DPCT on all *ISCAS '85* benchmark circuits from NanoSim^{TM} simulation as PS_{sim}, which is 84.4% as shown in column 8, row 12 of Table 6.2. Then, we calculate a using Equation 4.6. Then we apply this a to get the estimated active leakage power savings in Table 4.2.

$$a = PS_{sim}/PS_{est} = 0.978 \tag{4.6}$$

4.3.2.2 Computational Complexity of Finding the Optimal Partition

Different ways of combining the gates lead to different partition schemes. If there are N_{msw} groups of gates based on the MSW, the total number of possible different partitioning schemes N_{pt} can be calculated using the following Equation:

$$N_{pt} = \sum_{\{k_i,\ldots,k_m\}} \{C_{k_1}^{N_{msw}} \times C_{k_2}^{N_{msw}-k_1} \times \ldots \times C_{k_m}^{N_{msw}-k_{(m-1)}}\} \tag{4.7}$$

where $k_i,\ (i = 1,\ldots,m)$ are any integers satisfying $\sum_i^m k_i = N_{msw}$, and $C_{k_i}^{N_{msw}} = \binom{N_{msw}}{k_i}$.

For a big circuit, Npt_{total} can be a very huge number. So, it will be too computationally expensive to compare all possible partitioning schemes to get the optimal result. Actually, it is an NP-problem to find the optimal partition [15]. So, we propose a heuristic algorithm using dynamic programming to get the near-optimal partition with much less computational complexity.

4.3.2.3 Basic Ideas of the Heuristic Partitioning Algorithm by Dynamic Programming

To find a near-optimal solution for this partitioning problem, we need to reduce the search space as well as speed up the search process. Our heuristic algorithm applies two basic ideas to accomplish these two goals: 1) Reduce the search space using a heuristic. 2) Speed up the search process using dynamic programming.

First, our heuristic is: ***Given a set of MSWs, the partition that combines those MSWs that are next to each other is more likely to be the optimal partition***. For example, suppose we have

a simple set of MSWs: $(0,50)$, $(50,120)$, $(120,200)$. (Note: all of our timing windows are in *ps* units). Combining $(0,50)$ with $(50,120)$ will yield a new group whose switching window is $(0,120)$. However, combining $(0,50)$ with $(120,200)$ will yield a group whose switching window is $(0,200)$. Based on Equation 4.2, it is obvious that combining $(0,50)$ with $(50,120)$ is more likely to save more power than combining $(0,50)$ with $(120,200)$. At the same time, these two partitioning schemes both reduce the number of partitions by 1. So, their costs are similar. As a result, the first partitioning scheme, which combines nearby MSWs, is better. We say two MSWs are *nearby* if there are no other MSWs in between them.

Based on the above observation, we will only search those partitions that combine groups with nearby switching windows. Suppose that there are N_{msw} MSW groups initially. By only combining nearby groups, the number of possible partition schemes $P(N_{msw})$ is reduced to:

$$\begin{aligned} P(N_{msw}) = 1 &+ P(1)P(N_{msw}-1) + P(2)P(N_{msw}-2) + ... \\ &+ P(N_{msw}-2)P(2) + P(N_{msw}-1)P(1) \\ &- (N_{msw}-2) \end{aligned} \tag{4.8}$$

where $P(k)$ is the total number of possible partitions for a circuit with k MSWs when we only combine nearby groups. Obviously, $P(1) = 1$. So, we can derive $P(N_{msw})$ for any N_{msw} using this equation.

Even after reducing the search space by combining only nearby groups, the search space is still huge. So, we use dynamic programming to speed up the search process. The idea is to solve local small problems first and store the solutions. Then, we use these local small solutions to solve bigger problems. We continue doing this until the global problem is solved. By dynamic programming we trade memory with speed. The detailed dynamic programming algorithm is explained in the next section.

4.3.2.4 Flow of the Heuristic Partitioning Algorithm

The following is the detailed flow of our Heuristic partitioning algorithm:

1. *Round all original MSWs into integer picosecond units.*

   This reduces the total number of initial MSWs, as MSWs with differences that are less than $1ps$ can be considered as the same. Actually, based on the granularity one wants, one can round the initial MSWs to $5ps$ or $10ps$ to reduce the number even more.

Since each gate will have different MSWs if the delay calculation is in units of $0.1ps$ or $0.01ps$, the total number of initial MSWs will be almost equal to the total number of gates in the circuit, which can be very big. So, rounding MSWs can reduce the number of initial MSWs by many times, which will greatly reduce the size of the problem. The number of MSW groups of *ISCAS '85* benchmark circuits after rounding is shown in column 5 of Table 4.2. Compared with the number of gates in the circuit, which is also shown in the same table, this rounding process reduces the number of initial MSWs by 2-10 times.

Table 4.2: Heuristic Partitioning Results on *ISCAS '85* Benchmarks

Circuit	Number of Gates	Number of Levels	Worst Case Delay (*ps*)	# of Groups		Average # Gates per Group		Estimated Active Leakage Saving		Estimated Cost	
				Before Heuristic Partitioning	After Heuristic Partitioning	Before Heuristic Partitioning	After Heuristic Partitioning	Before Heuristic Partitioning	After Heuristic Partitioning	Before Heuristic Partitioning	After Heuristic Partitioning
c432	160	18	982	41	13	3.9	12.3	89.7%	88.8%	20.4%	11.1%
c499	202	12	855	13	10	15.5	20.2	81.3%	81.0%	12.4%	10.7%
c880	383	25	819	210	15	1.8	25.5	88.5%	80.8%	86.4%	14.2%
c1355	546	25	830	28	23	19.5	23.7	89.9%	89.2%	13.2%	11.5%
c1908	880	41	1024	367	17	2.4	51.8	91.9%	85.0%	107.6%	14.2%
c2670	1193	33	1467	431	23	2.8	51.9	92.5%	86.3%	98.2%	13.2%
c3540	1669	48	1647	747	17	2.2	98.2	92.5%	84.0%	169.4%	12.9%
c5315	2307	50	1515	778	14	3.0	164.8	91.6%	79.9%	209.9%	14.2%
c6288	2416	125	4547	868	40	2.8	60.4	97.4%	92.7%	97.4%	12.2%
c7552	3512	44	1258	1366	12	2.6	292.3	91.5%	74.6%	442.6%	15.5%
Average	1105.7	35.1	1245.3	484.9	18.4	5.7	80.8	90.7%	84.4%	125.8%	12.9%

2. *Sort all of the MSWs into non-decreasing order according to their start time.*

   As we want to combine nearby MSWs only, we have to sort them to line them up, so that we know which ones are nearby. Since each MSW has two parameters, the start time t and the end time T, MSWs that are nearby in start time may not be nearby in end time. Sorting by start time t may give totally different order from sorting by end time T.

   For example, suppose that we have three MSWs sorted by start time: $(50,300)$, $(60,120)$, $(70,150)$, where $(50,300)$ and $(60,120)$ are nearby and their combination will be tried in the search process. However, the order will be $(60,120),(70,150),(50,300)$ if sorted by the end time T, where $(50,300)$ and $(60,120)$ are no longer nearby and their combination will not be tried in the search process. So, a different ordering may lead to different results by our search algorithm. It is hard to say which is better, because the results may vary case by case. So, without losing generality, we just pick the start time

as the sorting standard. For groups with the same start time, the end time is then used for sorting.

Suppose that there are N_{msw} MSW groups left after the rounding process. First, we number them from 1 to N_{msw} according to the non-decreasing order of their start time. The group id number is used as the index for each MSW in the following steps. For example, group i refers to the MSW group whose group id is i. Second, we introduce a concept called $PartialCircuit_{i,j}$, which is the circuit including only the sorted MSW groups from i to j.

To save the corresponding start and end times of each MSW group, we create two two-dimensional arrays: $t[i][j]$ and $T[i][j]$, where $i, j = 1 : N_{msw}$; $t[i][j]$ and $T[i][j]$ are the start and end times of the switching window of the $PartialCircuit_{i,j}$ combining the MSW groups i to j. Note that $t[i][i]$ and $T[i][i]$ are just the start and end times of the original MSW group i.

Then, we create three two-dimensional arrays: $OPT[i][j]$, $PS[i][j]$ and $COST[i][j]$, where $i, j = 1 : N_{msw}$. These arrays are used to store the maximal OPT value and the corresponding PS and $COST$ values for this partitioning scheme of the $PartialCircuit_{i,j}$. As the purpose is to optimize OPT, we say a partitioning scheme for the $PartialCircuit_{i,j}$ is near-optimal when it yields the maximal $OPT[i][j]$.

We also create another two-dimensional array $Mark[i][j]$ to store the split position of the near-optimal partitioning for the $PartialCircuit_{i,j}$. For example, $Mark[i][j] = k$ means that the near-optimal partitioning of the $PartialCircuit_{i,j}$ is to split the $PartialCircuit_{i,j}$ at group k, from which we get $PartialCircuit_{i,k}$ and $PartialCircuit_{k+1,j}$. Then, we can trace down each individual $PartialCircuit_{i,k}$ and $PartialCircuit_{k+1,j}$ to get their corresponding near-optimal partition marks. Continuing this process, we can find all of the partitioning marks for the near-optimal partitioning of the $PartialCircuit_{i,j}$.

3. *Calculate the OPT of each individual group with the original MSW and record it in a table.*

   In this step, we assume that each individual MSW group is a partial circuit, which is just $PartialCircuit_{i,i}$, where $i = 1 : N_{msw}$. Since we know the number of gates in each group and their corresponding start and end times, we can use Equations 4.3, 4.4 and 4.5 to calculate the PS, $COST$ and OPT values for each $PartialCircuit_{i,i}$, which are $PS[i][i]$,

$COST[i][i]$ and $OPT[i][i]$. Here, the T_{cycle} we use is the cycle time of the entire circuit, not that of the partial circuit. Also, we set $Mark[i][i] = i$, which means that the near-optimal partitioning for each $PartialCircuit_{i,i}$ is itself. So, after this step, we get the near-optimal partitioning information of $PartialCircuit_{i,i}$, which is saved in $OPT[i][i]$, $PS[i][i]$, $COST[i][i]$ and $Mark[i][i]$, where $i = 1 : N_{msw}$.

4. *Calculate the OPT of each group that combines the two nearby MSWs. Compare it with the sum of two individual OPTs. Record the bigger one in a table as the near-optimal result and record the corresponding near-optimal grouping mark.*

   In this step, we will find the near-optimal partitioning for each $PartialCircuit_{i,i+1}$, where $i = 1 : N_{msw} - 1$. There are only two possible split mark positions to partition each $PartialCircuit_{i,i+1}$; the first position is $i+1$, which is to combine $PartialCircuit_{i,i}$ and $PartialCircuit_{i+1,i+1}$. The second possible position is i, which is to let the two partial circuits be separate. We then calculate the corresponding OPT values of the two partitioning schemes and pick the one with bigger OPT as the near-optimal partitioning for $PartialCircuit_{i,i+1}$.

   For the first scheme, the switching window will be ($min(t[i][i],\ t[i+1][i+1])$, $max(T[i][i],\ T[i+1][i+1])$). So, it is easy to calculate its PS, $COST$ and OPT values using Equations 4.3, 4.4 and 4.5. For the second scheme, we calculate its PS, $COST$ and OPT values using the following equations:

$$\begin{aligned} PS &= PS[i][i] + PS[i+1][i+1] \\ COST &= COST[i][i] + COST[i+1][i+1] \\ OPT &= OPT[i][i] + OPT[i+1][i+1] \end{aligned} \tag{4.9}$$

   Then, we pick the one with bigger OPT as the near-optimal $OPT[i][i+1]$. If the near-optimal one is the first, we set $Mark[i][i+1] = i+1$, otherwise, we set $Mark[i][i+1] = i$. We also update $PS_{[i][i+1]}$ and $COST_{[i][i+1]}$ using the corresponding values from the near-optimal partitioning way. So, after this step, we get the near-optimal partitioning information of $PartialCircuit_{i,i+1}$, which is saved in $OPT[i][i+1]$, $PS[i][i+1]$, $COST[i][i+1]$ and $Mark[i][i+1]$, where $i = 1 : N_{msw} - 1$.

5. *Calculate the OPT of each group that combines the three nearby MSWs. Compared it with the OPT of all other possible combinations of three MSWs using the available table.*

Record the biggest one as the near-optimal result in a table and record the corresponding near-optimal grouping mark.

In this step, we will find the near-optimal partitioning for each $PartialCircuit_{i,i+2}$, where $i = 1 : N_{msw} - 2$. There are three possible split positions to partition each $PartialCircuit_{i,i+2}$, which are $i+2$, $i+1$ and i.

The first way with split position as $i+2$ is just to combine $PartialCircuit_{i,i}$, $PartialCircuit_{i+1,i+1}$ and $PartialCircuit_{i+2,i+2}$. Similarly, the switching window of the combined group will be the ($min(t[i][i], t[i+1][i+1], , t[i+2][i+2]), max(T[i][i], T[i+1][i+1]), T[i+2][i+2])$). So, it is easy to calculate its *PS*, *COST* and *OPT* values using Equations 4.3, 4.4 and 4.5.

The second way with split position as $i+1$ is to combine $PartialCircuit_{i,i}$ with $PartialCircuit_{i+1,i+1}$, but let $PartialCircuit_{i+2,i+2}$ be separate. We calculate its *PS*, *COST* and *OPT* using the following equations:

$$\begin{aligned} PS &= PS[i][i+1] + PS[i+2][i+2] \\ COST &= COST[i][i+1] + COST[i+2][i+2] \\ OPT &= OPT[i][i+1] + OPT[i+2][i+2] \end{aligned} \tag{4.10}$$

The third way with split position as i is to combine $PartialCircuit_{i+1,i+1}$ with $PartialCircuit_{i+2,i+2}$, but let $PartialCircuit_{i,i}$ be separate. We calculate its *PS*, *COST* and *OPT* using the following equations:

$$\begin{aligned} PS &= PS[i][i] + PS[i+1][i+2] \\ COST &= COST[i][i] + COST[i+1][i+2] \\ OPT &= OPT[i][i] + OPT[i+1][i+2] \end{aligned} \tag{4.11}$$

Then, we pick the one with biggest *OPT* as the near-optimal $OPT[i][i+2]$. Based on which one is near-optimal, we update $Mark[i][i+2]$ with the corresponding split position. We also update $PS_{[i][i+2]}$ and $COST_{[i][i+2]}$ using the corresponding values from the near-optimal partitioning way. So, after this step, we get the near-optimal partitioning information of $PartialCircuit_{i,i+2}$, which is saved in $OPT[i][i+2]$, $PS[i][i+2]$, $COST[i][i+2]$ and $Mark[i][i+2]$, where $i = 1 : N_{msw} - 2$.

6. *Keep increasing the number of combined groups one by one until all MSWs are combined. Then we get the near-near-optimal OPT and the grouping marks of the near-optimal partitioning scheme for the entire circuit.*

   This is similar to Steps 4 and 5. Suppose that we want to find the near-optimal partitioning for each $PartialCircuit_{i,i+k}$, where $i = 1 : N_{msw} - k$. Then, there are will be $k+1$ possible split positions to partition each $PartialCircuit_{i,i+k}$, which are $i+k$, $i+k-1$, ... and i. similarly, we can calculate the *PS*, *COST* and *OPT* of each possible partitioning scheme using the results from previous steps. Then we choose the near-optimal one with maximal *OPT* and update $OPT[i][i+k]$, $PS[i][i+k]$, $COST[i][i+k]$ and $Mark[i][i+k]$ accordingly.

   If we keep doing this until we reach $PartialCircuit_{i,i+N_{msw}-1}$, where $i = 1$, we will get the near-near-optimal partition of the entire circuit. ***The result is near-near-optimal because we did not traverse the entire search space. We only tried the partitions that combine nearby groups. But it is an optimal result within the search space we traversed***.

4.3.2.5 Computational Complexity and Memory Complexity of the Heuristic Partitioning Algorithm

Based on the flow of the algorithm shown above, we can analyze the computational and memory complexity of our partitioning algorithm. The result is shown in Table 4.3. We can see that both the computational and memory complexity of our algorithm is $O(N_{msw}^2)$, where N_{msw} is the initial number of MSW groups in the circuit. N_{msw} is proportional to N, which is total number of gates in the circuit. But, N_{msw} is usually several times smaller than N due to our rounding process.

4.3.2.6 Experimental Results of the Partitioning Algorithm

Table 4.2 shows the number of groups, the number of gates per group, the estimated active leakage power saving and the related cost before and after heuristic partitioning. It also shows the number of gates and levels in each circuit. Before the heuristic partitioning, each individual MSW makes a group and the number of groups in each circuit is usually very large. By our heuristic partitioning, the average number of groups of a circuit reduces from 484.9 to 18.4

Table 4.3: Complexity of our Partitioning Algorithm

Step	Complexity	
	Computation	Memory
1: rounding	N_{msw}	N_{msw}
2: sorting	N_{msw}^2	$O(1)$
3: partitioning $PartialCircuit_{i,i}$	N_{msw}	N_{msw}
4-6: partitioning $PartialCircuit_{i,i+k}$ $k = 1,...,N_{msw}-1$	$\sum_{k=1}^{N_{msw}-1}(k+2)$	$\sum_{k=1}^{N_{msw}-1}(k+2)$
overall	$O(N_{msw}^2)$	$O(N_{msw}^2)$

and the average number of gates per group increases from 5.7 to 80.8, while the corresponding average cost reduces from 125.8% to 12.9%. At the same time the average active leakage power saving only changes from 90.7% to 84.2% after heuristic partitioning. Thus, our heuristic partitioning method reduces the average cost greatly with little effect on power savings.

4.3.3 3rd Step: Insert Cutoff MOSFETs

After heuristic partitioning of the circuit, a *p*MOS and an *n*MOS transistor are inserted to each group to control the V_{DD} and GND signals of the gates within that group. To minimize the extra delay caused by the cutoff MOSFETs, the size of cutoff MOSFETs has to be appropriate. If a power cutoff transistor is to control the power of a minimal sized inverter, it has been shown that the delay improvement becomes marginal beyond the size of $10\times$ of the minimal transistor width for the power cutoff transistor [12]. Also, not all gates switch at the same time within each group and many gates are complex CMOS gates. In our experiments, all transistors in the original circuit are minimal size. The widths of the cutoff control MOSFETs are set to be:

$$W = pw \times (10 \times L_{min}) \times n \tag{4.12}$$

where L_{min} is the minimum feature size in a given process technology, which is $70nm$ in our experiment; n is the number of gates within the group controlled by this cutoff control MOSFET; and $0 < pw < 1$ is the maximal percentage of gates switching at the same time within this group, which is related to the signal activities of PIs and the circuit's architecture. In our experiments, the signal activities of all PIs are chosen to be 0.5. Based on our experiments, we found the following empirical equations to set pw, which gives less than 6% delay penalty with

less than 15% average chip area cost.

$$\begin{aligned} pw &= 0.02 \quad \text{if} \quad n > 100 \\ pw &= 0.06 \quad \text{if} \quad 50 < n \leq 100 \\ pw &= 0.08 \quad \text{if} \quad 10 < n \leq 50 \\ pw &= 1/n \quad \text{if} \quad n \leq 10 \end{aligned}$$

4.3.4 4th Step: Generate Cutoff Control Signals

Cutoff control signals are used to control the power on/off of a group based on the switching window of that group. One pair of cutoff control signals is required for each group, one to control the cutoff *n*MOSFET and the other to control the cutoff *p*MOSFET. All cutoff control signals have the same period as the global clock signal. Suppose that the clock period is $1GHz$ with 50% duty cycle, and the MSW of a group (after heuristic partitioning) is $(60ps, 180ps)$. Figure 4.2 shows the waveforms of the clock and the two cutoff control signals for this group, $cutoff\text{-}cntr\text{-}n$ to control the cutoff *n*MOSFET and $cutoff\text{-}cntr\text{-}p$ to control the cutoff *p*MOSFET.

We use clock stretchers [89] to generate the power cutoff control signals for each group. An example clock stretcher used to generate the cutoff control signals in Figure 4.2 is in Figure 4.5. It has three inverters, a *NAND* gate and a C-switch. The signal $cutoff\text{-}cntr\text{-}n$ must rise at time $offset$ from the rising *clk* edge, and remain high for time *width*, so that its partition is powered at the correct time, relative to *clk*, so that the wavefront of signals passes through it using minimal energy. Variable Δ_i indicates the logic gate's incremental output delay in the clock stretcher from the rising clock edge. We size the inverters and *NAND* gate in the clock stretcher so that their delays satisfy these conditions:

$$\begin{aligned} width &= \text{MSW width} = (180-60)ps = 120ps \qquad (4.13) \\ &= \Delta_1 + \Delta_2 + \Delta_3 + \Delta_4 \\ &= t_{1f} + (t_{2r} - t_{2f}) + (t_{3f} - t_{3r}) + (t_{4f} - t_{4r}) \\ offset &= t_{2f} + t_{3r} + t_{4f} = 60ps \qquad (4.14) \end{aligned}$$

where t_{ir} (t_{if}) is the rising (falling) delay of gate i. For gate 1, an $INVERTER$, $t_{1f} = 120ps$. For gate 2, a *NAND* gate, $t_{2r} = 20ps$ is the best case rising delay and $t_{2f} = 20ps$ is the worst case falling delay. For $INVERTER$ 3, 4 and *transmission gate* 5, $t_{3f} = t_{3r} = t_{4f} = t_{4r} = t_{5f} =$

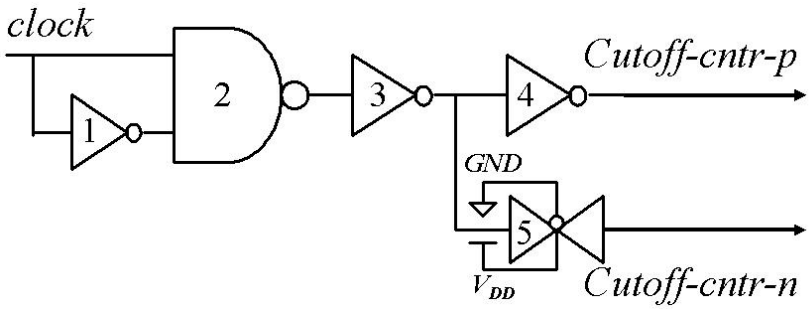


Figure 4.5: Clock Stretcher for Generating Cutoff Control Signals

$t_{5r} = 20ps$. Figure 4.2 shows the control signals for this example. If the input clock has skew Δ_{skew}, we need to replace the second equation to the following equqtion:

$$offset = t_{2f} + t_{3r} + t_{4f} - \Delta_{skew} = 60ps \tag{4.15}$$

We previously found 10% error in the static timing analysis compared with the analog simulator delay. We arbitrarily doubled the MSW for each gate, to make our method very insensitive to circuit delay variations due to various process corners. So, this allows up to 40% error in the rising and falling edge timings of cutoff control signals, so $delayerror = (10\% + 40\%) \times 2 = 100\%$. This greatly reduces the design complexity of the clock stretchers. If 1.5 times of MSW is used instead of 2.0 times, the allowed errors in the rising and falling edge timings of cutoff control signals will reduce to 15%. In both cases, analog simulation is used to verify the results to make sure that the cutoff control signals match our timing specifications. High V_{th} transistors should be used for all transistors in the clock stretchers to reduce their leakage power.

4.3.5 5th Step: Add Latches to POs to Capture the Data

With traditional power cutoff, data can get lost during the long sleep period due to the collapsed virtual V_{DD} and virtual GND signals. With DPCT, however, the power of each gate is only turned off for a short time within each clock cycle. Also each gate shares some power-on time with its fanout gates. So, data at each intermediate gate can be passed to its fanout gates correctly before it collapses. To capture the data on POs, we add a latch to each PO. The signal on each PO is stored in the latch right before we turn off the power of the gate that drives that PO. We use the power cutoff control signals of that gate to control the corresponding latch. In a real circuit where each PO is usually followed by a flip-flop, these latches can be removed.

4.3.6 6th Step: Verify the DPCT Circuit Using Analog Simulation

Finally, the circuit with DPCT is simulated using Cadence SPECTRE^{TM}. The results are compared with the SPECTRE^{TM} simulation results of the original circuit without DPCT under the same test vectors. All POs are checked one by one to make sure that the circuit with DPCT functions correctly. Our DPCT method was verified and proved to be working correctly on all *ISCAS '85* benchmarks.

4.4 Power Savings of DPCT

DPCT is mainly targeted for reducing active leakage power. However, it can also be used to reduce standby leakage power and dynamic power.

By turning on each gate only within a small part of the entire clock cycle, DPCT significantly reduces *active leakage power*. When the circuit is in standby mode, we can save *standby leakage power* by turning off the power connections of all groups. By turning on the power of a gate only within its switching window, the gate can make transitions only when all of its inputs are ready. This automatically balances the delay differences between the inputs of each gate. Therefore, glitches, which are unnecessary transitions of the output due to different delays on inputs, are automatically eliminated. This results in *dynamic power savings*. In our DPCT method, we multiple the width of MSWs by 1.5 up to 2.0 and combine the MSWs of some gates to reduce the extra cost of DPCT. A logic gate will have an output glitch if the path delays for an input transition from a PI to different inputs of the gate differ by an amount greater than the gate inertial delay. Combining MSWs of multiple gates, therefore, introduces glitches. But overall, circuits with DPCT have many fewer glitches compared with unmodified circuits, which may result in significant dynamic power savings. The experimental results of the power savings using DPCT are shown in Chapter 6.

Chapter 5

Power Grid and Process Variation Analysis on DPCT

5.1 Introduction

As the supply voltage and threshold voltage are decreasing with technology scaling, checking the integrity of the voltage on the power distribution network is becoming crucial. Since DPCT needs to turn the power connections of each group on and off at the system clock frequency, one concern is that it may disrupt the power grid by introducing more voltage drop and noise. Thus, we analyzed the effect of DPCT on the power grid. In Section 5.2, the procedure for the power grid analysis for DPCT is introduced [11]. The experimental results are shown in Chapter 6.

With technology scaling, process variations are posing an increasing challenge to the design, analysis and testing of nano-scale VLSI circuits. This is mainly due to the ever-increasing variabilities in the process parameters, such as channel length, transistor width, oxide thickness and the random placement of dopants in the channel. Since DPCT uses STA to calculate the switching window of each gate, process variations may have a big impact on its performance. To show how well DPCT will perform under process variations, we analyzed the effect of process variations on DPCT. In Section 5.3, the procedure for the process variation analysis on DPCT is introduced. The experimental results are shown in Chapter 6.

5.2 Power Grid Analysis on DPCT

To analyze the effects of DPCT on the power grid, we first map a circuit with DPCT to a power grid. Then, we map the same circuit without DPCT to the same power grid. Finally, we do analog simulations using $SPECTRE^{TM}$ on both circuits and compare the voltage drops on the two power grids.

5.2.1 Modeling of Power Grid

We model the power grid as a *RLC* network as shown in Figure 3.1. We set $R_{pg} = 0.1\Omega$, and $L_{pg} = 1pH$. The R_{pg} value is consistent with what is predicted by the *International Technology Roadmap for Semiconductors* (ITRS) [1], which is about 0.2Ω for a power grid branch whose pitch is $1\mu m$. The L_{pg} value is bigger than what is predicted by ITRS, which is about $0.02pH$. Here, the inductance in our experiments is calculated by scaling the value of Choi *et al.*'s modeling of a realistic on-chip power grid [22]. This larger L_{pg} helps us to study the Ldi/dt effects of DPCT as DPCT may result in relative bigger di/dt. To study the effect of decoupling capacitance on the power grid noise, we simulated different C_{pg} values including $0.01nF$, $0.1nF$, $1nF$ and $10nF$.

To map a circuit with DPCT to a power grid, we connect each group of gates to a power grid node. This is a legitimate practice since the average number of gates per group in a DPCT partition is about 100 and the pitch of a realistic power grid is about $1\mu m$. If $70nm$ or $45nm$ CMOS technology is used, one DPCT group can be conveniently fit into one power grid unit.

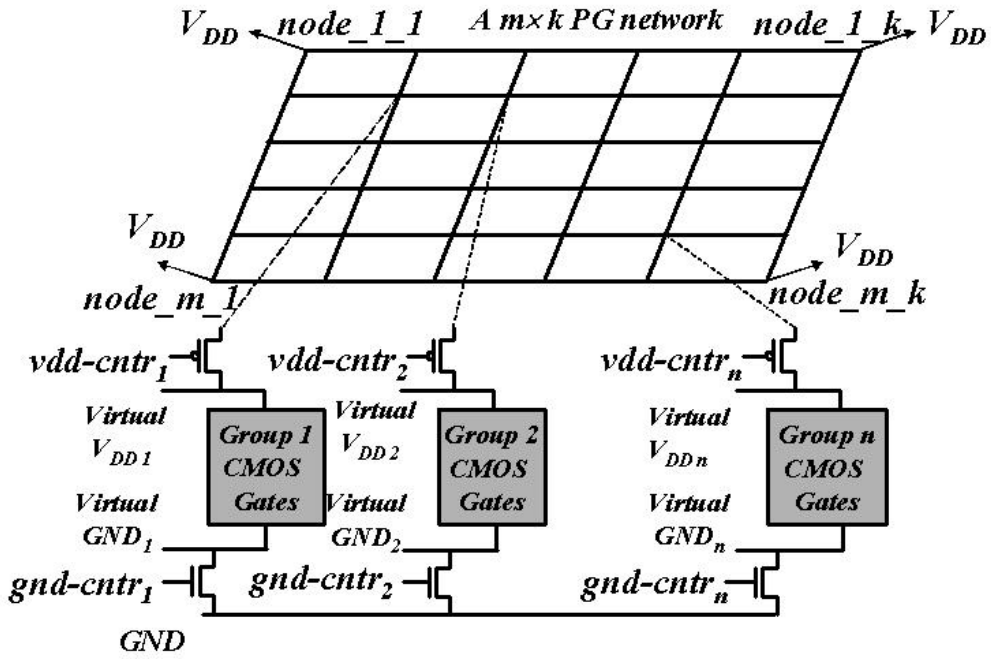


Figure 5.1: Mapping a DPCT Circuit to a Power Grid

Figure 5.1 shows how a circuit with DPCT is mapped to a power grid, which has *m* rows and *k* columns. The four nodes at four corners are connected to ideal V_{DD} as the variation in

the voltage levels among the pads is considered negligible as compared to the noise within the on-chip power distribution network. The other nodes are connected to the circuits such that each group of gates is connected to a node. Without losing generality, the size of the power grid is chosen to be a little bigger than the required number of nodes for connecting the circuits. We use *ISCAS '85* benchmark circuits for our power grid analysis. Table 5.1 shows the size of the power grid for each *ISCAS '85* circuit.

For a fair comparison between circuits with DPCT and without DPCT, we map the non-DPCT circuit to the same kind of power grid in the same way as the corresponding DPCT circuit, such that the same group of gates is connected to the same power grid node. Since the size of the power grids for *ISCAS '85* circuits are rather small, we use SPECTRE^{TM} to simulate the power grids.

Table 5.1: Power Grid Size of Each *ISCAS '85* Benchmark Circuit

Circuit	# of DPCT Groups	Average # of Gates per Group	Power Grid Size
c432	13	12	6×6
c499	10	20	6×6
c880	15	25	6×6
c1355	23	23	7×7
c1908	17	51	6×6
c2670	23	51	7×7
c3540	17	98	6×6
c5315	14	165	6×6
c6288	40	60	9×9
c7552	12	292	6×6

5.2.2 Procedures

The following is the procedure of our power grid analysis for DPCT:

1. *Generate SPECTRE netlists of* ISCAS '85 *circuits with the* power grid *(PG) network connected for both DPCT and non-DPCT*. This is done automatically by a C program.
2. *Analog simulation of the circuits using SPECTRE.*
3. *Collect the waveform data of all PG nodes of each circuit using an OCEAN script.*
4. *Analyze the data for all PG nodes of each circuit using a PERL script.*

5.3 Process Variation Analysis on DPCT

We know that DPCT partitions the circuits based on the MSW of each gate, which is calculated by STA. To analyze the effect of process variations on DPCT, we first do SSTA under process variations and calculate the MSW of each gate under each set of process parameters. Then we apply DPCT to the circuit for each set of MSWs. Finally, we estimate its power savings and compare them with the corresponding power savings without process variations.

5.3.1 Modeling of Process Variations

To model process variations, we consider transistor length L and width W, gate oxide thickness t_{ox} and threshold voltage V_{th} as independent normal distributed random variables [55]. We use truncated normal distributions for the above parameters to reflect the fact that the process variations in an operational chip cannot be more than some finite maximum value. We assume that their 3σ variations are 15% of their nominal values. Also, they are all truncated at 4σ, which is 20% of their nominal values. We also assume that all transistors have the same nominal values of L, W, t_{ox} and V_{th}, which are shown in Table 5.2

Table 5.2: Nominal Values for L, W, t_{ox} and V_{th}

	t_{ox} (m)	V_{th0} (V)	W (cm^{-9})	L (cm^{-9})
nMOSFET	1.6×10^{-09}	0.16	70	70
pMOSFET	1.7×10^{-09}	-0.19	70	70

To model the spatial correlations of intra-die process variations, we use the multi-level grid model [5] proposed by Agarawal *et al.* We use the gate's circuit level number as the indication of the gate's layout location. In this model, the area of the die is divided into regions using 3-level partitioning, as shown in Figure 5.2. For each level l, the die area is partitioned into 2^l squares, where the top level 0 has a single region and the bottom level 2 has 4 regions. Grid (i, j) refers to a grid that is the jth region on the ith level. The process variation of a transistor in any grid at the bottom level is then composed as the sum of the variations in that particular grid and the variations in all of its parent grids. For example, the variation of the channel length of transistors in grid $(2,3)$ is represented as:

$$\Delta W_{variation}(2,3) = \Delta W_{2,3} + \Delta W_{1,2} + \Delta W_{0,1} \tag{5.1}$$

where $\Delta W_{variation}(2,3)$ is the total variation in the width of all transistors in the grid $(2,3)$; $\Delta W_{2,3}$, $\Delta W_{1,2}$ and $\Delta W_{0,1}$ represent the variations in the widths in the grids $(2,3)$, $(1,2)$ and $(0,1)$, respectively.

Using this multi-level grid model, transistors that lie within closer proximity of each other will have more common intra-die variation components resulting stronger intra-die correlations. Also, the variations in grid $(0,1)$ model the inter-die variations since it is the parent grid of all other grids. So, both inter- and intra-die variations are represented using this model.

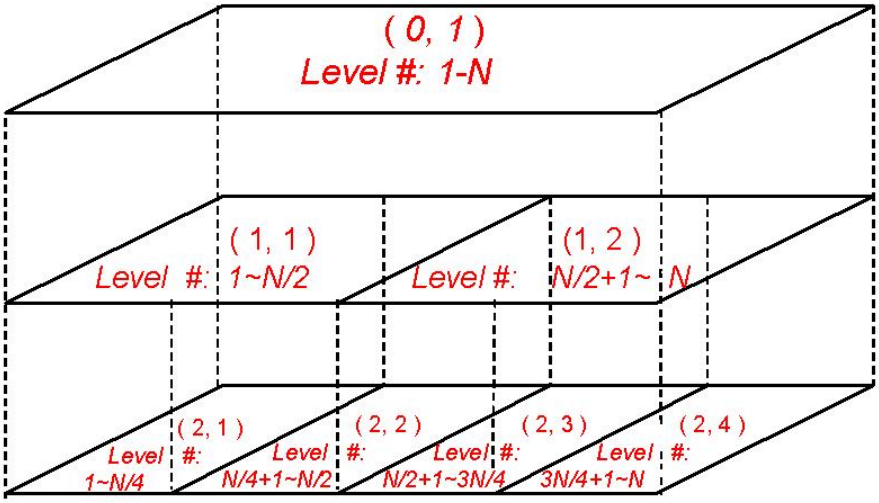


Figure 5.2: Modeling Spatial Correlations Using Quad-Tree Partitioning [5]

5.3.2 Modeling of Gate Delay

To estimate the delay of the circuit, we model each CMOS gate as an *RC* network, which is the same approach as used by Wei *et al.* [86]. The load capacitance *C* is calculated using the parameters and equations defined in the BSIM3v3 model manual [46]. A look-up table based on SPECTRE^{TM} analog simulations is used to get the equivalent *R* of the *n*-tree or *p*-tree of a CMOS gate based on the gate type, the number of fanins, the number of fanouts and the transistor sizes to compute the equivalent on-resistance. The delay calculation results from static timing analysis were verified on various benchmark circuits to be within 10% error compared with the results of SPECTRE^{TM} analog simulation. The delay calculation, static timing analysis and MSW calculation are implemented as C programs.

5.3.3 Procedures

The following is our procedure for analyzing the effect of process variations on DPCT:

1. *Model the inter- and intra-die process variations using the multi-level grid model.*

   To model process variations, we use truncated normal distributions for transistor length L and width W, gate oxide thickness t_{ox} and threshold voltage V_{th}. We assume that they are all independent of each other, and their 3σ variations are 15% of their nominal values. We pick 4σ to be the truncation threshold, which is 20% of their nominal values.

   We assume that the V_{th} for both nMOSFETs and pMOSFETs are random variations that have no dependence on location of devices. L, W and t_{ox} are spatially correlated variations, which are modeled using the multi-grid model. We use the 3 layer 1×4 grid to model the spatial correlations of these parameters. Then, we use Equation 5.1 to calculate the overall variations of each parameter. We also normalize the overall variations so that their 3σ variations are 15%.

2. *Do SSTA using the Monte Carlo method.*

   Without losing generality, we use the Monte Carlo method to do SSTA, which is the most accurate SSTA technique. For each *ISCAS '85* benchmark circuit, we first use a Gaussian random generator to generate $10,000$ samples of ΔV_{th} of both nMOSFETs and pMOSFETs with the desired distributions. Then we use the same Gaussian random generator to generate $10,000$ samples of ΔL, ΔW and Δt_{ox} for each level of grid. Then, we use Equation 5.1 to calculate the overall variations of L, W and t_{ox}. After normalizing and truncating each parameter, we do STA for the circuit at each set of variations to get the MSW for each gate.

   The random generator we used is called the R250 Gaussian random number generator [2], which is based on the uniform random generator algorithm proposed by Kirkpatrick and Stoll [41]. The correctness of this random generator is verified using MATLAB.

3. *Estimating the power savings of DPCT under process variations*.

   After we do STA for each set of variations and get the MSWs for each gate, we partition the circuit using our heuristic partitioning algorithm for each set of MSWs. Then we apply DPCT to the circuit for each set of variations, and estimate the active leakage power savings of DPCT using Equation 4.5.

4. *Summarize the statistical distribution of the power savings under process variations.*

   For the 10,000 samples of variations, we get 10,000 samples of active leakage power savings. Then we can calculate the mean and standard deviation of the power savings under process variations. By comparing the DPCT power savings with and without process variations, we get the effect of process variations on DPCT.

5.4 Summary

In this chapter, we introduced the procedures for doing two important analyses for DPCT: the power grid analysis and the process variation analysis. The power grid analysis is to study the effect of DPCT on the power grid. Analog simulations with $SPECTRE^{TM}$ are used to analyze the effect of DPCT on the power grid, which is modeled with a *RLC* model. The process variation analysis is to study the effect of process variations on DPCT. Monte Carlo method is used for the process variation analysis. Both analyses are critical for the practical application of DPCT. The results of the analysis are given in Chapter 6.

Chapter 6

Results

6.1 Experimental Results for Power Savings

We tested DPCT on the *ISCAS '85* benchmarks in a 70*nm* CMOS process modeled by Berkeley Predictive Models [17]. Table 6.1 shows some of the key parameters of the transistor model we used in our experiments, where N_{ch} is the peak channel doping concentration and N_{gate} is the poly-gate doping concentration. Here, $N_{ch} = N_A$ for *n*MOSFETs, and $N_{ch} = N_D$ for *p*MOSFETs.

For each benchmark circuit, the circuit without DPCT and the one with DPCT are running at the same frequency using the same test vectors. Random test vectors with 0.5 activities are used for all of the PIs. The clock period of the test vectors for each benchmark is chosen to be an integer about 10% larger than the worst-case circuit delay. V_{DD} is set to $1.0V$. The temperature is set to 90°C to reflect the real chip temperature when the circuit is active. Single low V_{th} MOSFETs are used, where the V_{th} voltages are $0.16V$ and $-0.19V$ for *n*MOSFETs and *p*MOSFETs, respectively. All circuits are simulated using Synopsys Nanosim^{TM} (an analog circuit simulator) with 70*nm* analog transistor models for the logic gates to get their detailed power profile. The results are shown in Table 6.2. Synopsys Nanosim^{TM} counts both short-circuit power and leakage power as wasted power. Also, the short circuit power is included in the active leakage power part in Table 6.2.

Table 6.1: BSIM3v3 Model Parameters of the 70*nm* CMOS Process by Berkeley Predictive Models

	t_{ox} (m)	V_{th0} (V)	N_{ch} (cm^{-3})	N_{gate} (cm^{-3})
*n*MOSFET	1.6×10^{-09}	0.16	$1.0 \times 10^{+18}$	$5.0 \times 10^{+20}$
*p*MOSFET	1.7×10^{-09}	-0.19	$1.0 \times 10^{+18}$	$5.0 \times 10^{+20}$

6.1.1 Power Savings for DPCT

DPCT saves up to 90% of the active leakage power, up to 54% of the dynamic power and up to 72% of the total power. The average active leakage saving is 84.4%, the average dynamic power saving is 9.7% and the average overall power saving is 40.1%. The power savings of DPCT on bigger circuits are more significant than on smaller circuits. As operating cutoff transistors introduce extra dynamic power, the dynamic power saving will be negative if the dynamic power saved by reducing glitches is smaller than the extra cost. That is why the dynamic power savings are small or negative on relatively small circuits, but quite significant on larger circuits such as c6288, where glitches are much more significant than in any other benchmark.

When the circuit is in standby mode, we can save standby leakage power by turning off the power to all groups. Our experimental results on *ISCAS '85* benchmark circuits show more than 99% average standby leakage power savings.

Table 6.2: Power Savings and Area Cost of DPCT on ISCAS '85 Benchmarks

Circuit	Clock Frequency (Hz)	Total Power			Active Leakage Power			Dynamic Power		
		No DPCT (μW)	With DPCT (μW)	Savings	No DPCT (μW)	With DPCT (μW)	Savings	No DPCT (μW)	With DPCT (μW)	Savings
c432	$1G$	75.06	50.44	32.8%	35.76	21.73	80.6%	39.30	43.51	−10.7%
c499	$1G$	179.39	111.93	37.6%	100.05	21.73	78.3%	79.34	90.20	−13.7%
c880	$1G$	140.72	114.13	18.9%	65.09	10.81	83.4%	75.63	103.31	−36.6%
c1355	$1G$	209.83	151.51	27.3%	101.39	15.93	84.3%	108.44	135.51	−24.9%
c1908	$800M$	345.59	242.75	29.8%	141.27	22.98	83.7%	204.32	219.76	−7.6%
c2670	$625M$	495.85	275.57	44.4%	240.80	29.27	87.8%	255.05	246.30	3.4%
c3540	$500M$	508.20	273.83	46.1%	310.90	42.10	86.5%	197.30	231.73	−17.5%
c5315	$625M$	1064.60	625.57	41.2%	509.00	88.64	82.6%	555.60	536.93	3.4%
c6288	$200M$	837.42	237.85	71.6%	453.85	59.94	86.8%	383.58	177.91	53.6%
c7552	$625M$	1600.42	793.69	50.4%	725.21	72.95	89.9%	875.20	720.74	17.7%
Average		545.71	287.73	40.1%	268.33	37.13	84.4%	277.38	250.59	9.7%

6.1.2 Power Effeciency Improvements for DPCT

Dynamic power is used to charge and discharge the load capacitances in the circuits. So, it is essential for the proper functioning of the circuits. However, the active leakage power (including short-circuit power) is totally useless for the normal functioning of the circuits. So, it is called wasted power. We define the *Power Efficiency* (PE) of a circuit as the ratio of the dynamic power $P_{dynamic}$ to the total power P_{total}, which is shown in Equation 6.1. PE is the percentage of power used for the useful transitions of the circuits. The higher PE means less power is wasted.

$$PE = P_{dynamic}/P_{total} \tag{6.1}$$

DPCT mainly reduces active leakage power, which is the wasted power. By reducing more than 80% of the wasted power, DPCT increases the circuits' power effeciencies dramatically. Figure 6.1 shows the power effeciencies of *ISCAS '85* benchmark circuits with and without DPCT. Without DPCT, the average power effeciency of *ISCAS '85* benchmark circuits is just 50.4%. With DPCT, the average power effeciency increases to 86.3%.

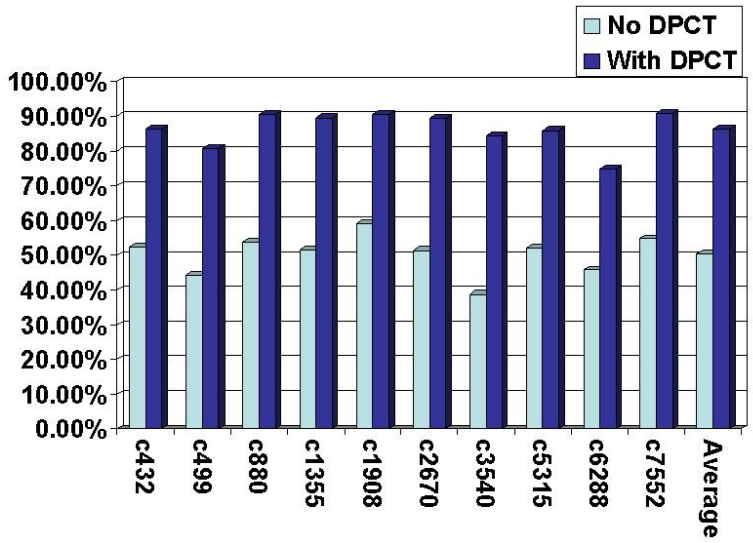


Figure 6.1: Power Efficiencies of *ISCAS '85* Benchmarks with and without DPCT

6.1.3 Effect of MSW Window Size on Power Savings of DPCT

In the above experiments, we doubled the width of the MSW to be $((T_o - D) - 0.5 \times D, T_o + 0.5 \times D)$ to allow for the delay estimation error and process variations. As the power savings of DPCT are given by Equation 4.5, the change of the MSW width will change the $Width_k$ parameter in that equation, which will result in changed power savings. To study the effect of different MSW widths on power savings, we experimented with timing windows that were 1.0, 1.1, 1.2, 1.3, 1.4, 1.5 and 2.0 times the MSW width. The 1.0, 1.1 and 1.2 figures gave output logic errors because there is not enough overlap between the power-on times of nearby groups. The 1.3 and 1.4 values only work for some of the benchmarks. But, the 1.5 and 2.0 values worked correctly on all benchmarks. Table 6.3 shows the minimal working MSW window size for each benchmark circuit and the corresponding power savings. Overall, the average total power saving increases 7% compared with the circuits using the 2.0$\times$ MSW. To choose the

appropriate MSW widths for a circuit, we can start from 1.5 times and try 1.4 and 1.3 until logic errors happen.

Table 6.3: Minimal MSW Size of *ISCAS '85* Circuits and the Corresponding Power Savings

Circuit	Minimal MSW Window	Total Power			Active Leakage Power			Dynamic Power		
		No DPCT (μW)	With DPCT (μW)	Savings	No DPCT (μW)	With DPCT (μW)	Savings	No DPCT (μW)	With DPCT (μW)	Savings
c432	1.3×	75.06	46.36	38.24%	35.76	5.36	85.01%	39.30	41.00	−4.33%
c499	1.3×	179.39	88.82	50.49%	100.05	18.44	81.57%	79.34	70.38	11.29%
c880	1.4×	140.72	105.22	25.23%	65.09	6.94	89.34%	75.63	98.28	−29.95%
c1355	1.3×	209.83	146.66	30.11%	101.39	15.13	85.08%	108.44	131.53	−21.29%
c1908	1.4×	345.59	214.28	38.00%	141.27	16.86	88.07%	204.32	197.42	3.38%
c2670	1.3×	495.85	237.51	52.10%	240.80	19.84	91.76%	255.05	217.67	14.66%
c3540	1.4×	508.20	246.15	51.56%	310.90	31.56	89.85%	197.30	214.59	−8.76%
c5315	1.5×	1064.60	517.12	51.43%	509.00	62.70	87.68%	555.60	454.42	18.21%
c6288	1.4×	837.42	169.25	79.79%	453.85	27.28	93.99%	383.58	141.97	62.99%
c7552	1.5×	1600.42	746.92	53.33%	725.21	91.69	87.36%	875.20	655.23	25.13%
Average				47.02%			86.11%			19.87%

6.1.4 Delay and Area Cost of DPCT

There are two costs of DPCT, delay and chip area. Just as with other power cutoff techniques, DPCT introduces about 6% delay. To minimize the delay, the power cutoff MOSFETs usually are more than 10 times larger than other transistors. Clock stretchers, used to generate cutoff control signals, also add extra chip area. These altogether introduce 15% area overhead, on average. The area of the circuit is calculated as the sum of the sizes of all transistors in the circuits. Table 6.4 gives the area overhead of DPCT on *ISCAS '85* benchmarks.

Table 6.4: Area Cost of DPCT on *ISCAS '85* Benchmarks

c432	c499	c880	c1355	c1908	c2670	c3540	c5315	c6288	c7552	Average
29.1%	12.1%	20.2%	23.3%	16.3%	13.7%	9.2%	6.0%	6.0%	5.2%	14.9%

6.2 Experimental Results of Power Grid Analysis

We analyzed DPCT's effect on the power grid using *ISCAS '85* benchmarks in a 70*nm* CMOS process modeled by Berkeley Predictive Models. For each benchmark circuit, the circuit without DPCT and the one with DPCT are running at the same frequency using the same test vectors. Random test vectors with 0.5 activities are used for all of the *primary inputs* (PIs). The

power supply V_{DD} is set to $1.0V$. The temperature is set to 90°C to reflect the real chip temperature when the circuit is active. Single low V_{th} MOSFETs are used, where the V_{th} voltages are $0.16V$ and $-0.19V$ for *n*MOSFETs and *p*MOSFETs, respectively.

6.2.1 A Typical Power Grid Node for DPCT and non-DPCT Circuits

Figure 6.2 shows the waveforms of a power grid node *node_3_3*, which is the node on the 3rd row and 3rd column, of c432 without DPCT and with DPCT. We can see that the maximal voltage drop on the power grid node of the DPCT circuit is smaller than that of the non-DPCT circuit. However, there are more fluctuations on the DPCT power grid node compared to those of the non-DPCT circuit. (*Note that the voltage drops across both circuits are very small due to the small current of each group of gates in c432.*)

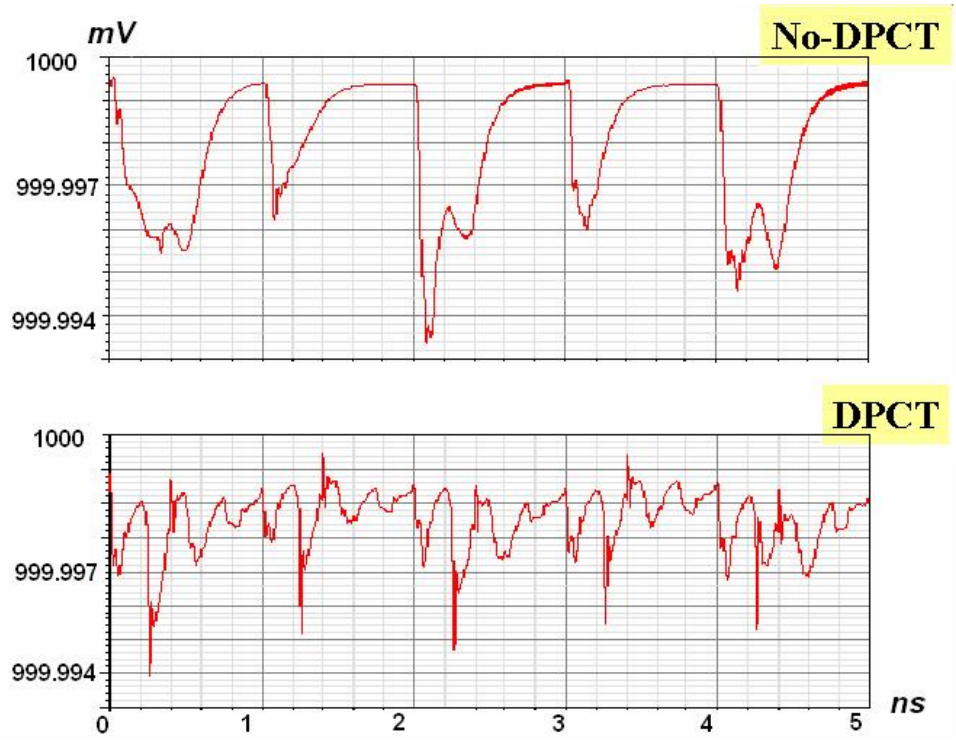


Figure 6.2: Typical Waveform of a Power Grid Node

6.2.2 Spectral Analysis of Power Grid Nodes

To get a deeper understanding of the effect of DPCT on the power grid, we did FFT spectral analysis on the two power signals, which is shown in Figure 6.3. We can see that the maximal component of the non-DPCT circuit is at 1*GHz*, which is equal to the clock frequency. But, the maximal component of the DPCT circuit is at 4*GHz*. However, the absolute value of the maximal component of the DPCT circuit is 51% smaller than that of the non-DPCT circuit. The harmonic noise at 5*GHz* of DPCT is only 2.7% bigger than for the non-DPCT circuit. Overall, DPCT has smaller components at all frequencies except 4*GHz* and 5*GHz*. This means that DPCT reduces the maximal voltage drop on this power grid node. But it also increases it a little for some high frequency harmonic noise. Extra decoupling capacitors can be used to reduce the high frequency harmonic noise to the required level.

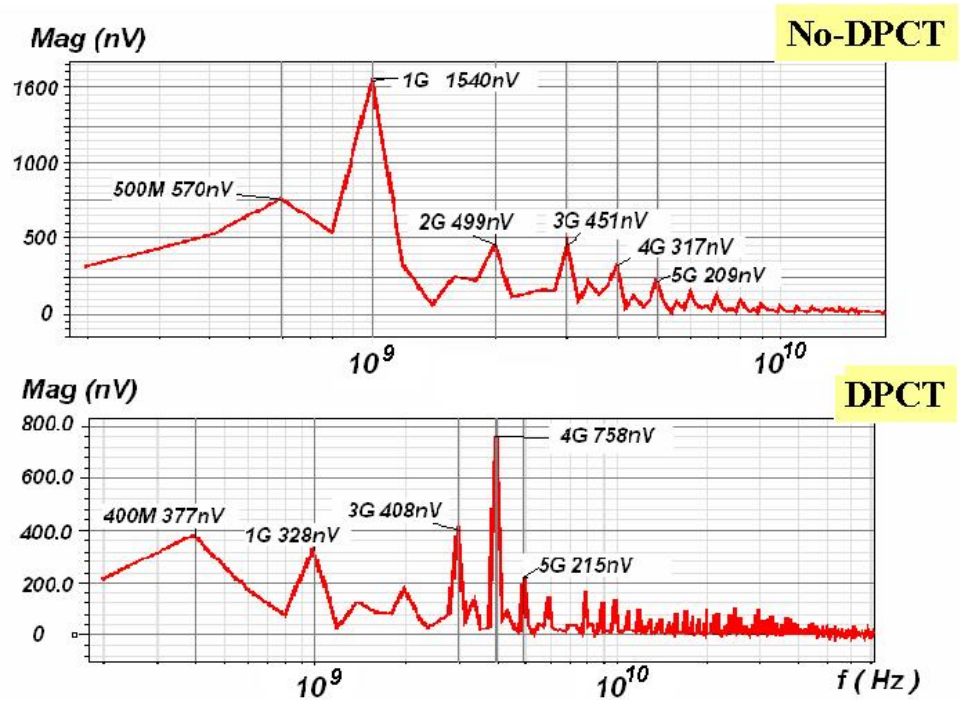


Figure 6.3: The Spectrum of a Power Grid Node

6.2.3 Maximal Voltage Drop on All Power Grid Nodes

To analyze the overall power grid integrity, we compare the maximal voltage drop of all power grid nodes between the DPCT circuits and non-DPCT circuits. Table 6.5 shows the results

for different C_{PG} values, which are the capacitances of power grid branches. We can see that the maximal voltage drop of a DPCT circuit is 30%-39% smaller than that of the non-DPCT circuit, on average.

Table 6.5: Maximal Voltage Drop on *ISCAS '85* Benchmarks

Circuit	Maximal Voltage Drop on All Power Grid Nodes								
	$C_{pg} = 0.01nF$			$C_{pg} = 0.1nF$			$C_{pg} = 1nF$		
	No DPCT (μV)	With DPCT (μV)	Reduction	No DPCT (μV)	With DPCT (μV)	Reduction	No DPCT (μV)	With DPCT (μV)	Reduction
c432	23	20	13.0%	11	8	27.3%	7	6	14.3%
c499	56	27	51.8%	30	14	53.3%	20	10	50.0%
c880	43	43	0.0%	23	21	8.7%	14	13	7.14%
c1355	50	44	12.0%	22	19	13.6%	15	14	6.7%
c1908	81	79	2.5%	51	36	29.4%	27	20	25.9%
c2670	102	79	22.6%	70	40	42.9%	42	26	38.1%
c3540	126	89	29.4%	86	52	39.5%	46	27	41.3%
c5315	227	107	52.9%	160	71	55.6%	92	40	56.5%
c6288	156	55	64.7%	94	35	62.8%	59	26	55.9%
c7552	318	164	48.4%	243	107	56.0%	135	61	54.8%
Average			29.7%			38.9%			35.1%

To understand why DPCT can reduce the maximal voltage drop on the power grid, we compare the waveform of the total current of a DPCT circuit with the corresponding non-DPCT circuit, which is shown in Figure 6.4. We can see that the maximal current of non-DPCT circuit c6288 is $3.5mA$, while the maximal current of DPCT circuit c6288 is just $1.6mA$, which is 54.3% smaller. Thus, the maximal *IR* drop of a DPCT circuit will be much smaller. Since *IR* drop is the dominant part of the voltage drop on power grid, this reduction in the maximal current demand results in the reduction of the maximal voltage drop on the power grid network. Meanwhile, there is more turbulence on the current waveform of the DPCT circuit c6288 due to the cutoff operations, which explains why DPCT may increase some high frequency harmonic noise.

The big reduction in the maximal current demand is due to the fact that DPCT turns on the power of the circuit group by group in a sequential order. This is similar to the approach to turn on the circuit part by part to reduce the current spike when we power up a big circuit.

6.2.4 Maximal Voltage Drop vs. C_{PG}

We simulated different C_{PG} values to see the effect of the decoupling capacitance on the power grid noise. Figure 6.5 shows the relationship between the maximal voltage drop and C_{PG} for

both the DPCT and non-DPCT c432 circuits. We can see that increasing C_{PG} reduces the maximal voltage drop on the power grid. So, we can add more decoupling capacitance on the power grid to meet the desired noise target, which is a standard practice in VLSI power grid design. Since DPCT circuits have smaller maximal voltage drop than non-DPCT circuits with the same C_{PG}, a smaller C_{PG} can be used for DPCT circuits to meet the same noise target than that of non-DPCT circuits. This will reduce the overall area overhead of DPCT.

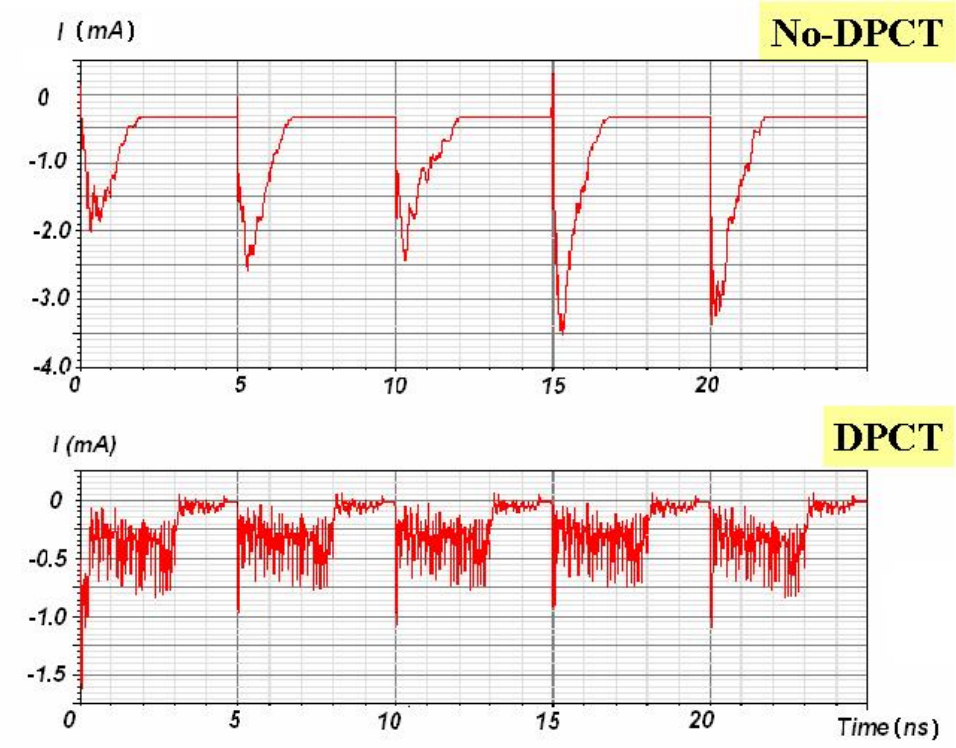


Figure 6.4: Total Current of c6288 without DPCT and with DPCT

6.2.5 Summary of Power Grid Analysis

We analyzed the effect of DPCT on the power grid. Experimental results show that DPCT can reduce the maximal voltage drop on the power grid. At the same time, DPCT may slightly increase the high frequency harmonic noise of the power grid. Adding some extra decoupling capacitance on the power grid nodes can reduce the power grid noise to the desired level.

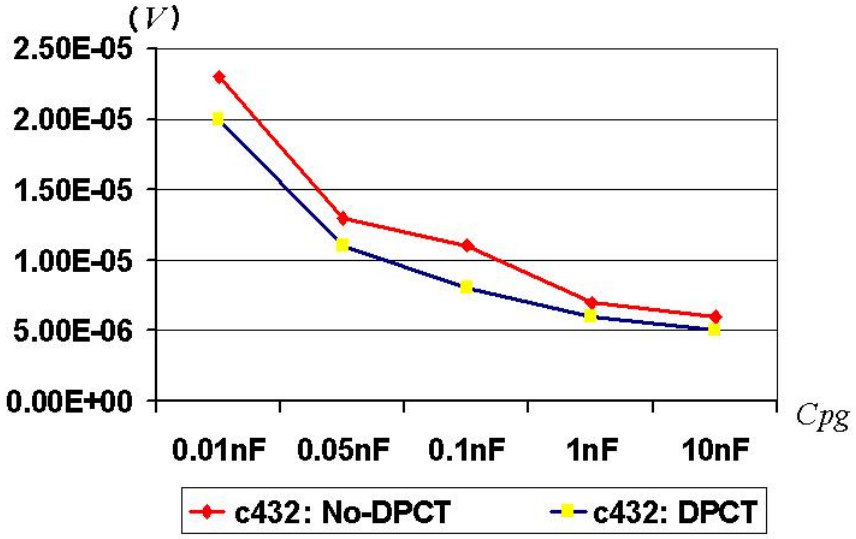


Figure 6.5: Maximal Voltage Drop versus C_{PG}

6.3 Experimental Results for Process Variation Analysis

We analyzed both the inter- and intra-die process variations' effect on DPCT power savings using *ISCAS '85* benchmarks in a 70*nm* CMOS process modeled by Berkeley Predictive Models. To model process variations, we consider transistor length L and width W, gate oxide thickness t_{ox} and threshold voltage V_{th} as independent normal distributed random variables [55]. We use truncated normal distributions for the above parameters to reflect the fact that the process variations in an operational chip cannot be more than some finite maximum value. We assume that their 3σ variations are 15% of their nominal values. Also, they are all truncated at 4σ, which is 20% of their nominal values. Please refer to Table 5.2 for the nominal value of each parameter.

For each benchmark circuit, 10,000 samples were used for the SSTA using the Monte Carlo method. DPCT is then applied to each set of benchmark circuits and its power savings are estimated. Finally, we calculate the statistical distributions of the clock cycles, timing windows and power savings and compare them with the nominal results, which are just the values without process variations.

6.3.1 Process Variations' Effect on Clock Cycles

Under process variations, the 3σ value is usually used for the clock cycle, which is calculated as $mean(T_{cycle}) + 3\sigma_{T_{cycle}})$, where T_{cycle} is the clock cycle. Overall, 99.7% of the samples fall within the 3σ region. Based on our simulation results, the clock cycles of each benchmark circuit also are normal distributions. Figure 6.6 shows the histograms of the clock cycles of c6288 and c7552 with 10,000 samples. From the figures, we can see that they all distribute as normal distributions.

To get a clear picture of the effects of process variations on the clock cycles, we compare the 3σ clock cycle with the nominal clock cycle. We also show the worst-case clock cycles, which are the clock cycle values when all of the transistor lengths L and widths W, gate oxide thicknesses t_{ox} and threshold voltages V_{th} are set to be the worst-case values. For L, t_{ox} and V_{th}, the worst-case values are 20% larger than the nominal values. However, the worst-case value of W is 20% smaller than the nominal value. Table 6.6 shows the comparisons of the nominal, 3σ and worst-case clock cycles of *ISCAS '85* benchmark circuits. We can see that the 3σ clock cycle is 52.6% larger than the nominal clock cycle on average. This means that process variations will reduce the clocking rate by a third, on average. At the same time, the worst-case clock cycle is 17.8% larger than the 3σ clock cycle on average. So, we will lose 20% more on speed if we use worst-case corner analysis instead of the statistical timing analysis. So, statistical timing analysis has to be used for CMOS circuit deisgn under process variations.

Table 6.6: Clock Cycles (*ps*) of *ISCAS '85* Benchmark Circuits under Process Variations

Circuit	Nominal	Mean	σ	3σ Value	3σ Value Increase over Nominal	Worst-case	Worst-case Increase over 3σ Value
c432	982.0	1009.4	74.6	1233.2	25.6%	1491.8	21.0%
c499	855.0	910.5	75.0	1135.6	32.8%	1358.7	19.6%
c880	819.0	853.2	65.4	1049.4	28.1%	1304.9	24.3%
c1355	830.0	876.9	70.4	1087.9	31.1%	1314.1	20.8%
c1908	1024.0	1818.6	99.8	2118.0	106.8%	2336.6	10.3%
c2670	1467.0	2130.2	122.3	2497.0	70.2%	2861.3	14.6%
c3540	1647.0	2686.2	144.1	3118.4	89.3%	3458.6	10.9%
c5315	1515.0	1927.2	119.3	2285.0	50.8%	2679.6	17.3%
c6288	4547.0	4701.0	365.8	5798.4	27.5%	7225.7	24.6%
c7552	1258.0	1755.5	102.3	2062.5	64.0%	2370.8	15.0%
average					52.6%		17.8%

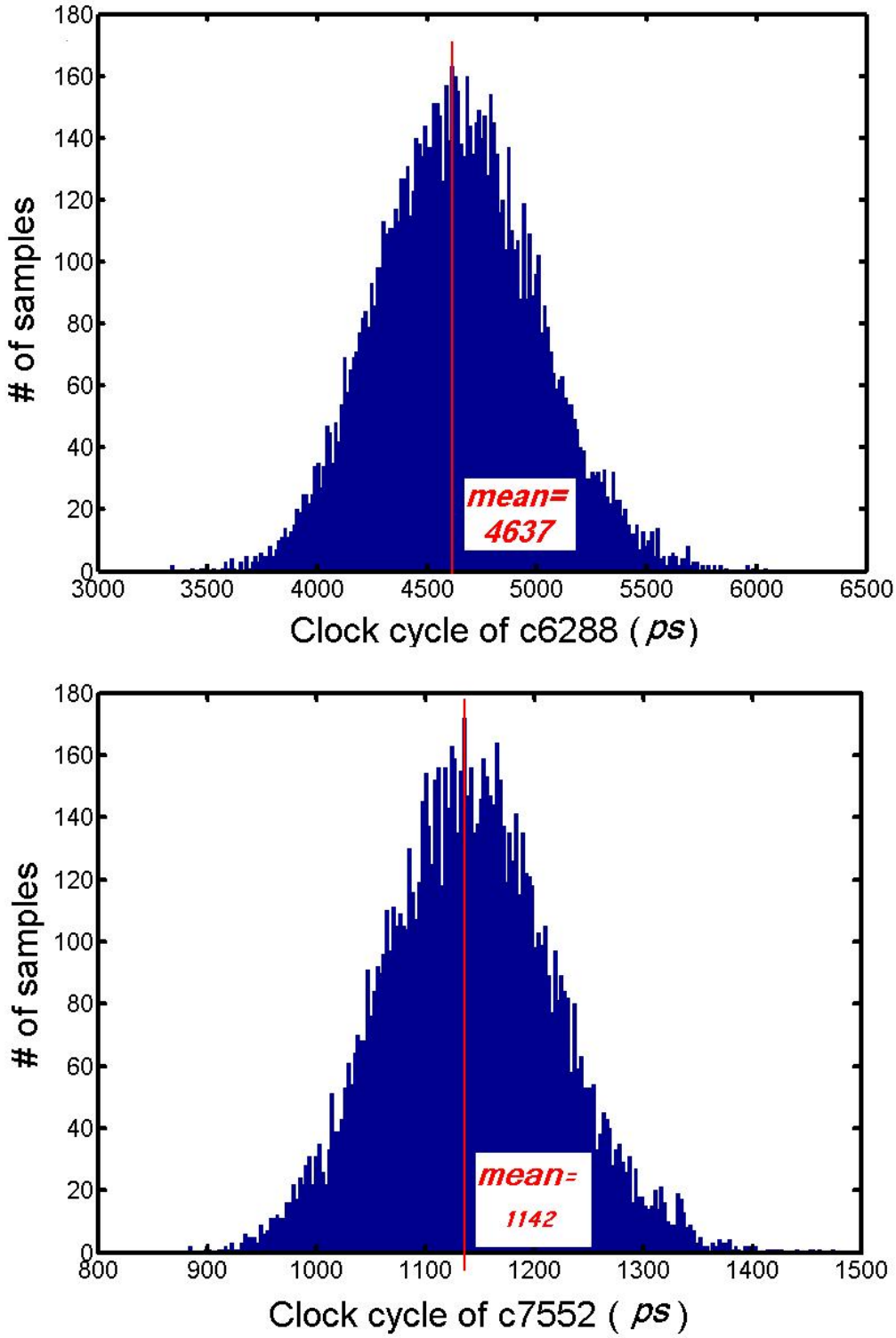


Figure 6.6: Histograms of the Clock Cycles of c6288 and c7552

6.3.2 Process Variations' Effect on Cutoff Windows

Under process variations, the switching window (t, T) is calculated as $(mean(t) - 3\sigma_t,\ mean(T) + 3\sigma_T)$. For the clock cycle T_{cycle}, we use the 3σ value, i.e., $(mean(T_{cycle}) + 3\sigma_{T_{cycle}})$. Although both the cutoff window width and the clock cycle increase under process variations, the process variations have the effect of both reducing the start time and increasing the end time of all MSWs. However, only the end time of the clock cycle is increased by the process variations. So process variations will increase the cutoff window width and reduce the ratio of the power-off time to the clock cycle of the circuit. Columns 3 and 4 in Table 6.7 show the average ratio of the power-off window width to the 3σ clock cycle of each *ISCAS '85* benchmark circuit. We can see that the ratio reduces by 11.5% on average.

Table 6.7: Process Variations' Effect on DPCT with *ISCAS '85* Benchmarks

Circuit	# of gates	Ratio of Power-off Window to 3σ Clock Cycle			Estimated Active Leakage Power Saving			Estimated Total Power Saving		
		Nominal	With Process Variations	Reduction	Nominal	With Process Variations	Reduction	Nominal	With Process Variations	Reduction
c432	160	0.91	0.85	7.20%	80.5%	74.8%	7.1%	32.8%	30.0%	8.4%
c499	202	0.89	0.74	16.86%	78.2%	64.4%	17.6%	37.6%	29.9%	20.4%
c880	383	0.91	0.76	15.54%	83.5%	71.6%	14.3%	18.9%	13.4%	29.1%
c1355	546	0.95	0.91	4.52%	84.3%	80.0%	5.2%	27.8%	25.7%	7.6%
c1908	880	0.92	0.79	14.15%	83.2%	72.1%	13.3%	29.6%	25.0%	15.3%
c2670	1193	0.94	0.85	10.24%	87.8%	74.7%	14.9%	44.4%	38.1%	14.3%
c3540	1669	0.92	0.78	15.18%	86.5%	70.1%	18.9%	46.1%	36.1%	21.7%
c5315	2307	0.90	0.79	12.64%	82.6%	69.5%	15.8%	41.2%	35.0%	15.1%
c6288	2416	0.97	0.95	2.23%	86.9%	85.1%	2.1%	71.6%	70.7%	1.4%
c7552	3512	0.87	0.72	16.90%	89.9%	73.6%	18.1%	50.4%	43.0%	14.6%
Average	1106	0.92	0.81	11.54%	84.3%	73.6%	12.7%	40.0%	34.7%	14.8%

6.3.3 Process Variations' Effect on Power Savings

The increase of the ratio between the cutoff window width and 3σ clock cycle will reduce the active leakage power savings. Using Equation 4.5, we can estimate the active leakage power savings due to DPCT. Columns 6 and 7 in Table 6.7 show the estimated active leakage power savings without or with process variations. We can see that the average active leakage power saving reduces from 84.3% to 73.6%, down by 12.7%, which is shown in column 8 in Table 6.7.

From the work of Yu and Bushnell [10], we also know the percentage of active leakage power and dynamic power compared to the total power of each benchmark circuit. If we assume

that the dynamic power savings are not changed by process variations, we can estimate the total power changes caused by process variations, which are shown in columns 9, 10 and 11 in Table 6.7. We can see that the average total power saving reduces from 40.0% to 34.7%, down by 14.8%.

6.3.4 Conclusions on Process Variation Analysis

Process variations are becoming a major factor that lowers the performance of VLSI CMOS circuits. In this thesis, we analyzed the effect of process variations on DPCT. We found that process variations reduce the average active leakage power savings and total savings of DPCT by 12.7% and 14.8%. Although the loss of performance is signficant, DPCT still gives excellent power savings under process variations, and saves 73.6% of the average active leakage power and 34.7% of the average total power.

Chapter 7

A Layout Implementation of DPCT

7.1 Introduction

In the previous chapter, we experimented with DPCT using ideal transistor level *ISCAS '85* benchmark circuits. In practical circuit design, ideal transistor level netlists are usually used in the early stage of the design process. Finally, all circuits will be implemented at the layout level and verified using the extracted netlists from layouts. The extracted netlists have all of the parasitic devices of the layout, such as parasitic capacitances, resistances and inductances. The parasitic devices, especially the parasitic capacitances, are the main differences between the real layout level circuits and the ideal transistor level circuits. With these parasitic devices, the layout level circuits usually perform worse than the ideal transistor level ones. So, the simulation results from the circuits extracted from the layout are more accurate predictions of the circuit performance. Thus, it is better to implement DPCT in layout level to further verify its performance.

In this chapter, we present a layout implementation of a 16-bit multiplier and c432, an interrupt controller, with DPCT. Both the layout design process and the simulation results are given. Some issues regarding the layout design of DPCT circuits are also discussed. We also give some background knowledge about the ASIC and custom-design flow in the beginning.

7.2 Background on *Application-Specific Integrated Circuit* (ASIC) and Custom-Design Flow

A design flow is a set of procedures that allows designers to progress from a specification for a chip to the final chip implementation in an error-free way [89]. Most chip designs fall into two kinds of design flows: ASIC and full custom-design. The ASIC design flow offers high productivity for most large digital chips with moderate performance requirements. But the full custom-design flow is used for smaller analog, RF and high-speed digital chips that require

higher performance, or for very high volume chips, such as microprocessors and cell phones.

7.2.1 ASIC Design Flow

An ASIC is an *integrated circuit* (IC) customized for a particular use, rather than intended for general-purpose use. The normal ASIC design always starts from a *specification* and goes through the following steps before mass production: *logic design*, *physical design* and *prototype*. The detailed ASIC design flow is shown in Figure 7.1.

1. *Specification*. The specification gives the detail design requirements for the system, such as speed, power, size, cost, etc. It gives the guidelines for the following system design process.

2. *Logic Design*. The logic design usually uses high level hardware description languages, such as Verilog and VHDL, to model the system. After simulation and verification, it will be synthesized into a gate level netlist. The gate level netlists use some basic logic gates in a certain standard cell library to implement the circuits, such as NAND, NOR, XOR, INVERTER, etc. The gate level netlist will be simulated and verified again before the physical level design.

3. *Physical Design*. The physical design usually uses standard cell libraries to map the gate level netlists to layouts. The standard cell library is a collection of layout implementations of the logic gates used in the gate level netlists. Automated placement and routing tools are available to do the mapping and routing automatically. Then, the post-layout netlists are extracted from the layouts to simulate and verify the circuit again.

4. *Prototyping*. Before mass production, a small number of sample chips are fabricated for system testing. This is to fully test the system as some effects in the circuits may not be captured in the previous layout-level verifications. If the design meets the specification after prototyping, the design process is done and the chip is ready for mass production.

7.2.2 Custom-Design Flow

A custom-design flow is used for high-performance chips, such as microprocessors, analog and RF chips, etc. The custom-design also starts from a specification and goes through the logic design, physical design and prototyping steps. Manually drawn schematics are used in the logic

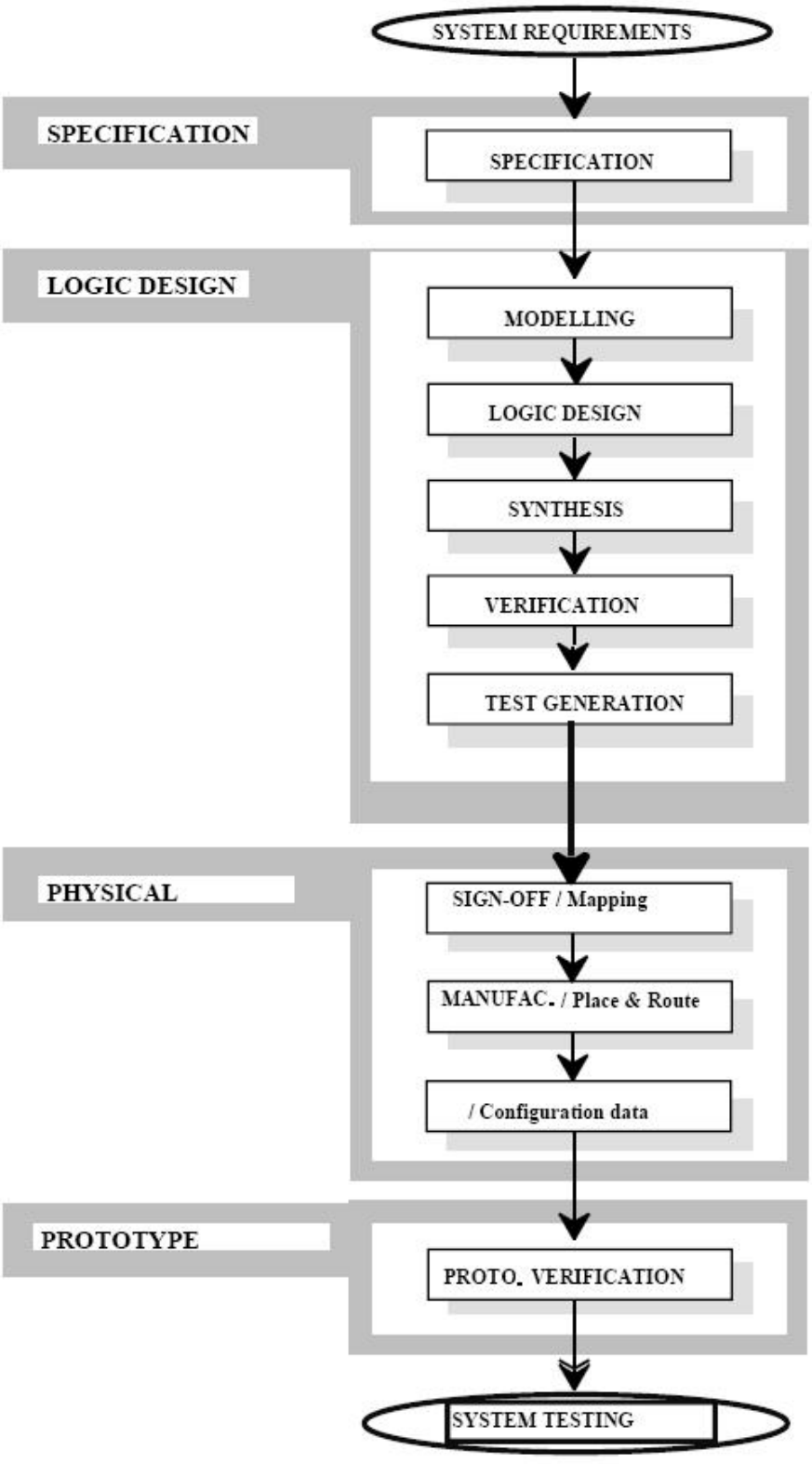


Figure 7.1: ASIC Design Flow

design step, rather than using standard cell based synthesis from Verilog or VHDL. Also, hand-crafted layouts are used for the physical implementation of the circuit after the schematics are simulated and verified.

7.3 Physical Implementation of a 16-bit Multiplier with DPCT

As we discussed above, physical design is just one step in the chip design process. Its main purpose is to map the gate level netlists into layouts. Although standard cell based design is widely used in ASICs, hand-crafted design is usually used for physical design of high performance systems, including ALUs. There are two main reasons for this. First, the standard cell based design does not give good performance on ALUs. Second, ALUs are highly modularized circuits and can be fairly easily designed by hand due to their high regularity in structure. For example, a 16-bit adder can be built using multiple blocks of 1-bit adders. Since the 16-bit multiplier has the same structural modularity and regularity, we also implemented its layout by hand. For comparisons, we build the layouts for both the multiplier with and without DPCT. Then, we simulate, verify both designs and compare their performance. In the following, we give the details of the architecture of the 16-bit multiplier and its physical design process.

7.3.1 Architecture of the 16-bit Multiplier

A multiplier is one of the key components in the ALU, which is a digital circuit that calculates arithmetic operations (such as addition, subtraction, multiplication, etc.) and logic operations (such as AND, OR, XOR, etc.) between two numbers. A 16-bit multiplier multiples two 16-bits operands and generates a 32-bit result. A multiplier is usually implemented with adders because multiplication is just the addition of multiple partial products. For illustration, Figure 7.2 shows a basic architecture of a 4-bit multiplier. We can see that it is made of two kinds of basic units: a 1-bit *carry save adder* (CSA) and a 1-bit *carry ripple adder* CPA. The CSA unit, which is shown as the square box, is equal to an adder that adds up the partial product AB, with carry-in C_{in} and S_{in}. It generates sum S_{out} and carry-out C_{out}. The CSAs are organized and connected in the special way as shown in the figure, such that each CSA's S_{out} will be connected to the S_{in} of the CSA that is below it. While, each CSA's C_{out} will be connected to the C_{in} of the CSA that is on its bottom left. Finally, the S_{out} and C_{out} from the bottom level of CSAs are feed into the CPAs to get the final product results. The CPA unit, which is shown as the square box with round corners, is just the normal full adder that adds up A, B and C_{in} without the AND gate calculating AB. Each CPA unit is also organized in a special way such that the C_{out} output

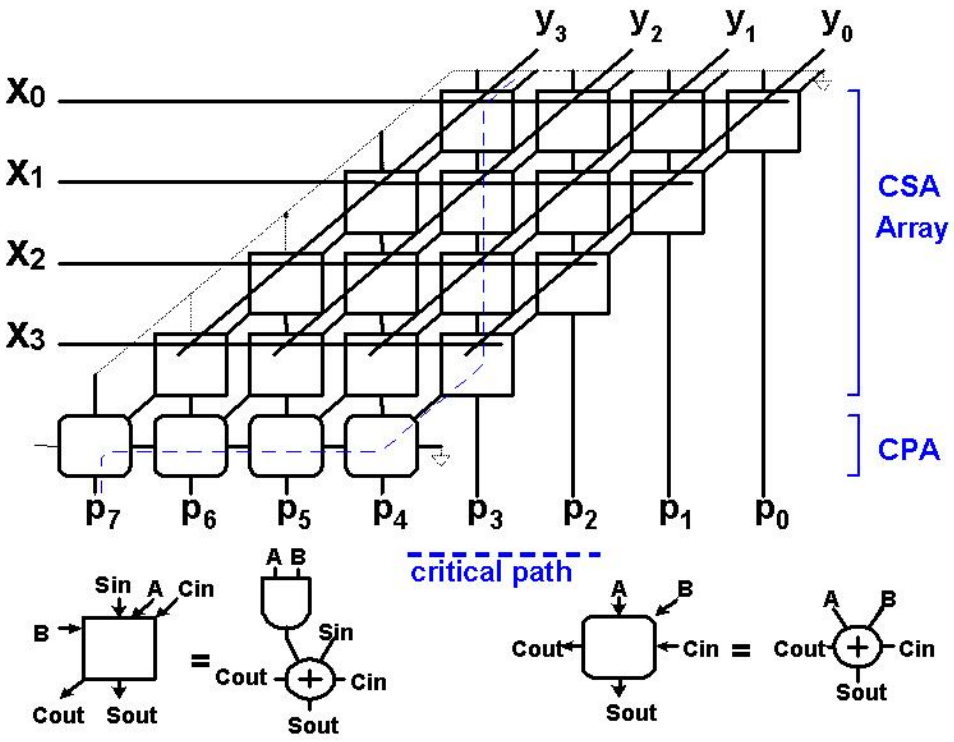


Figure 7.2: The Architecture of a 4-bit Multiplier

of each CPA directly contacts with the C_{in} input of the CPA unit on its right. Overall, the 4-bit multiplier needs a 4 × 4 CSA array and a 4-bit CPA. The critical path of the multiplier is the path to propagate the S_{out} through 4 CSA units and C_{out} through 4 CPA units.

A 16-bit multiplier can be implemented using a similar architecture. It will consist of a 16 × 16 CSA array and a 16-bit CPA. The critical path will be the path to propagate the S_{out} through 16 CSA units and C_{out} through 16 CPA units. To speed up the multiplier, Booth-encoding is usually used to reduce the number of partial products. To simplify the design, we only implemented the regular multiplier without Booth-encoding.

7.3.1.1 Layout Design of the 16-bit Multiplier without DPCT

Based on the architecture of the 16-bit multiplier we discussed above, we implemented its layout by hand. To do this, we first designed the layouts of the 1-bit CSA and 1-bit CPA units. The input and output ports of the CSA and CPA layouts are carefully positioned such that they can be easily lined up as shown in Figure 7.2. A small adjustment is made such that the final layout of the 16-bit multiplier forms as a rectangle. Figure 7.3 shows the schematic of the full adder we used in our CSA and CPA units. Figures 7.4 and 7.5 show the layouts of the 1-bit

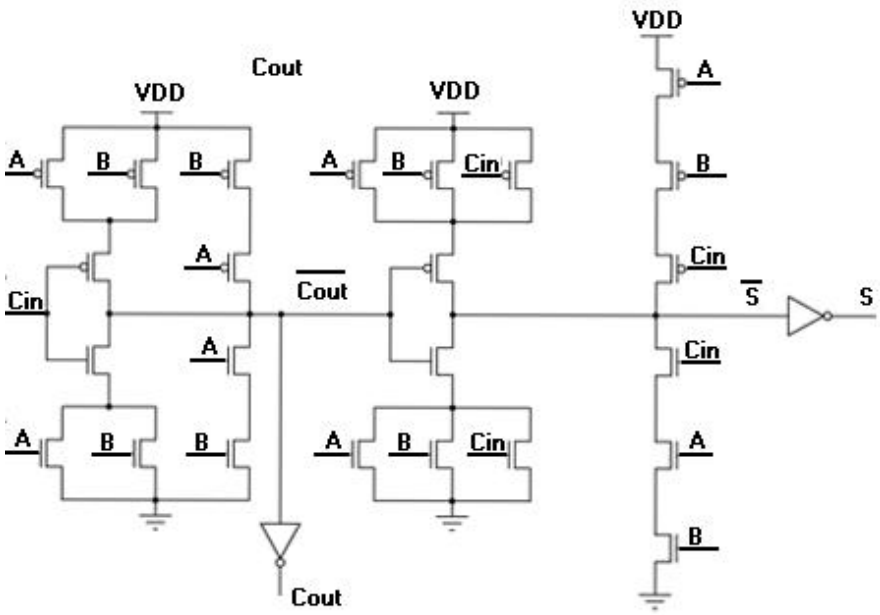


Figure 7.3: The Schematic of a Full Adder

CSA and 1-bit CPA. Figure 7.6 shows the full layout of the 16-bit multiplier without DPCT. Although the final layout of the 16-bit multiplier looks very complicated, it is fairly easy to design using the two basic building blocks: the 1-bit CSA and the 1-bit CPA.

7.3.1.2 Procedures for the Layout Design of the 16-bit Multiplier with DPCT

The following is the procedure for the physical design of the 16-bit multiplier with DPCT:

1. *Partition the Multiplier*. To apply DPCT to the 16-bit multiplier, we first need to partition the multiplier into multiple groups. We could use the algorithm described in Chapter 4 to do the partitioning. However, due to the regular structure of the multiplier, we can easily partition the multiplier based on its modular structure. From Figure 7.2 we can see that each row of the CSA array makes a group, whose switching window width is the propagation delay of the S_{out} of a CSA unit. The start time of the switching window for each CSA group is shifted by the delay of C_{out} of a CSA unit. There are 16 rows of CSA arrays, which make 16 DPCT groups: groups 0 to 15. Also, the row of 16 CPAs makes a group, group 16, whose switching window is the propagation delay of the C_{out} of 16 CPA units. The start time of the switching window of the CPA group is further shifted by the delay of the C_{out} of a CSA unit from group 15. Overall, there are will be 17 groups in this circuit: groups 0 to 16. The architecture of the 16-bit multiplier with DPCT is shown

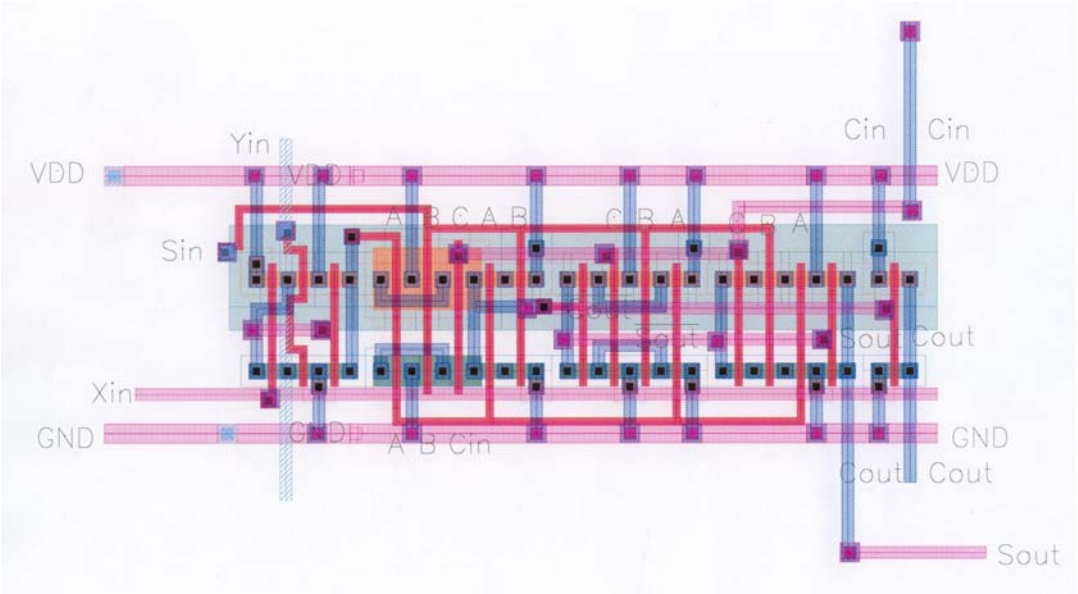


Figure 7.4: The Layout of a 1-bit CSA

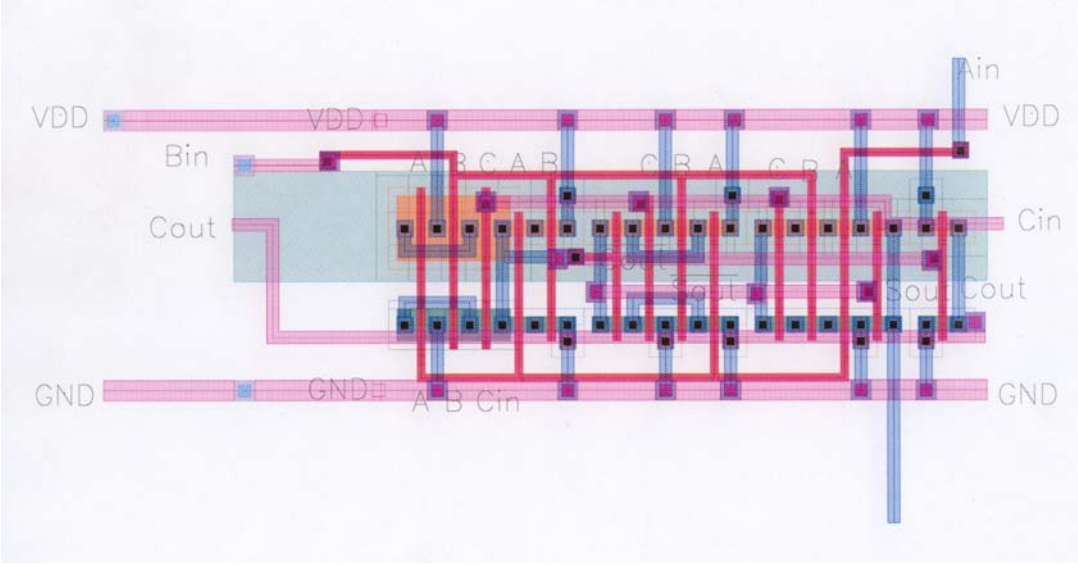


Figure 7.5: The Layout of a 1-bit CPA

Figure 7.6: The Layout of the 16-bit Multiplier without DPCT

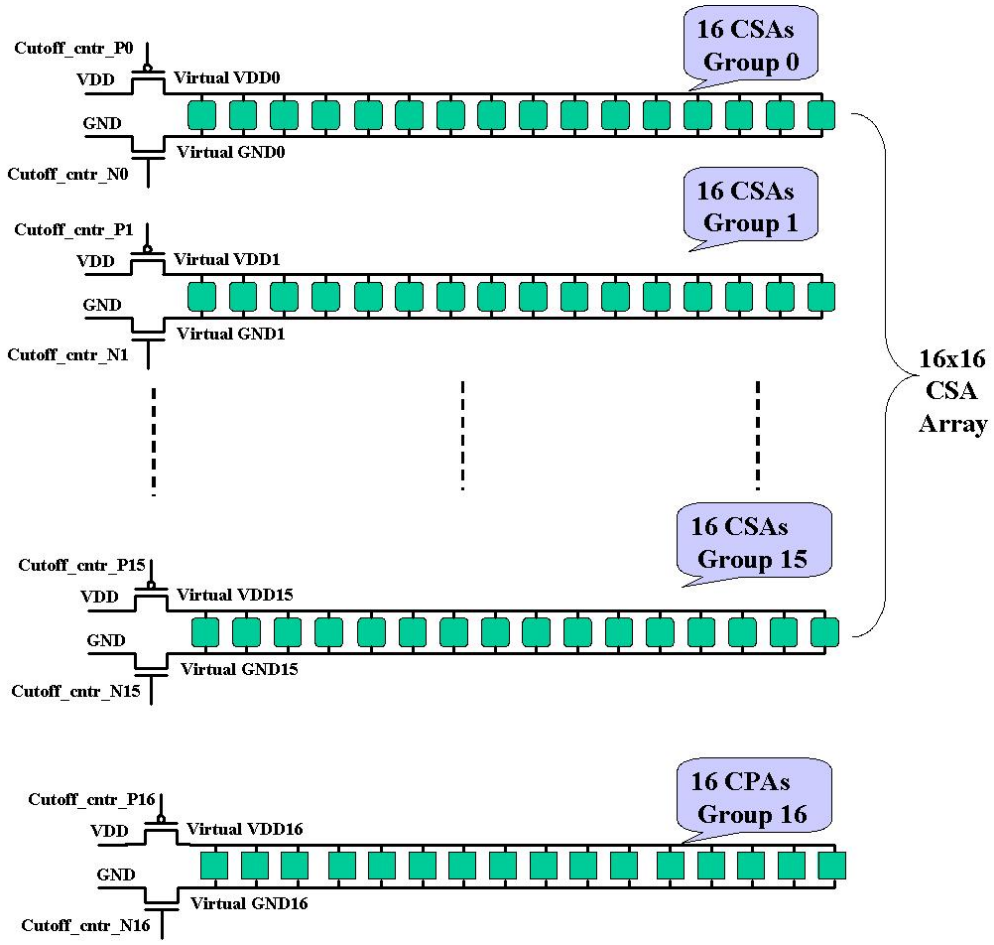


Figure 7.7: The Architecture of the 16-bit Multiplier with DPCT

in Figure 7.7. A pair of cutoff MOSFETs and two cutoff control signals are required for each group.

2. *Calculate the Switching Window of each DPCT Group*. We already know that the propagation delay of S_{out} and C_{out} of the 1-bit CSA and CPA units defines the switching window width and offset of each DPCT group. To get these delays, we do analog simulations on the 1-bit CSA and CPA using $SPECTRE^{TM}$. After that, we get all of the timing information required for the cutoff signals of each DPCT group. We also double the original switching window width to allow 50% overlap between nearby DPCT groups. This also covers the delay increase incurred by the cutoff transistors. So, no further adjustments in the partitioning will be needed, which saves much design effort. Figure 7.8 shows the clock and the *p*MOSFET cutoff control signals for the first four DPCT groups. The clock frequency is $300MHz$. The pulse width of each cutoff control signal is about $260ps$, and the shifting offset relative to its previous group is about $120ps$. The overlap of the on-time of two nearby groups is $260ps - 120ps = 140ps$. All of the cutoff control signals of groups 0 to 15 are of the same width and shifting offset as these four signals. Since group 16 will be on at the second half of the clock cycle, we use the system clock and its inverse signal as the cutoff control signals for this group. This further simplifies the design without sacrificing the performance.

3. *Layout Design of the Multiplier with DPCT*. We first implement the DPCT version of the layouts for each CSA and CPA unit. In the DPCT version layout, the power of the each gate becomes virtual *VDD* and virtual *GND*. The substrate contacts of each unit are connected to global *VDD* and *GND*. So, there will be two sets of power lines on the layout. Figure 7.9 shows the DPCT version layout of the 1-bit CSA. We can see that it has two extra power lines for the substrate contacts, which are *P-SUB-CONTACT* and *N-SUB-CONTACT*, respectively. They will be connected to the global *VDD* and *GND* accordingly. The *VDD* and *GND* in this layout are actually virtual *VDD* and virtual *GND*, which will be connected to the global *VDD* and *GND* through power cutoff transistors.

   With the DPCT version layouts for the 1-bit CSA and CPA, we can build the layout of the 16-bit multiplier following a similar structure as shown in Figure 7.2. We also add two power cutoff transistors and a clock generator to each group. Note that the shifting offsets

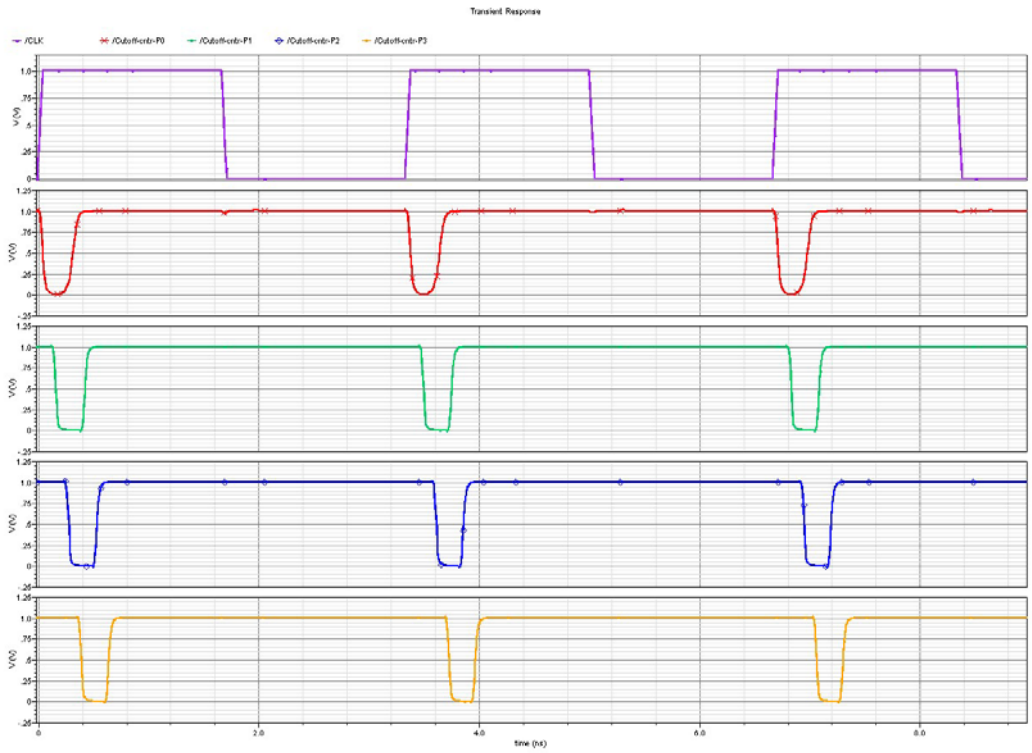


Figure 7.8: The System Clock and *p*MOSFT Cutoff Control Signals for Groups 0 to 3

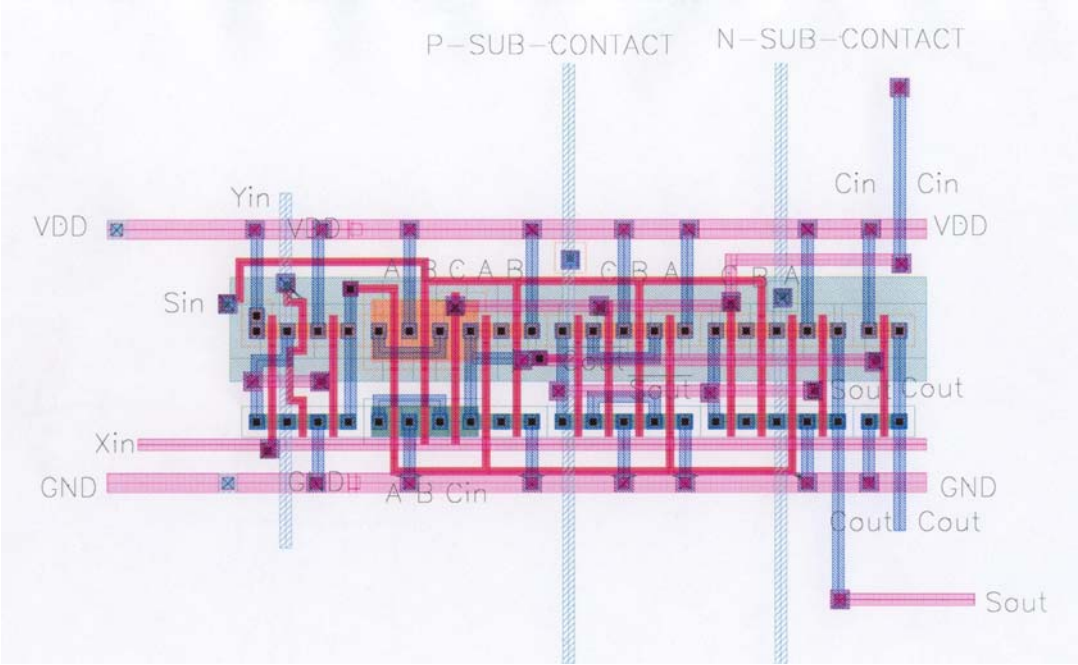


Figure 7.9: The Layout of a 1-bit CSA for DPCT

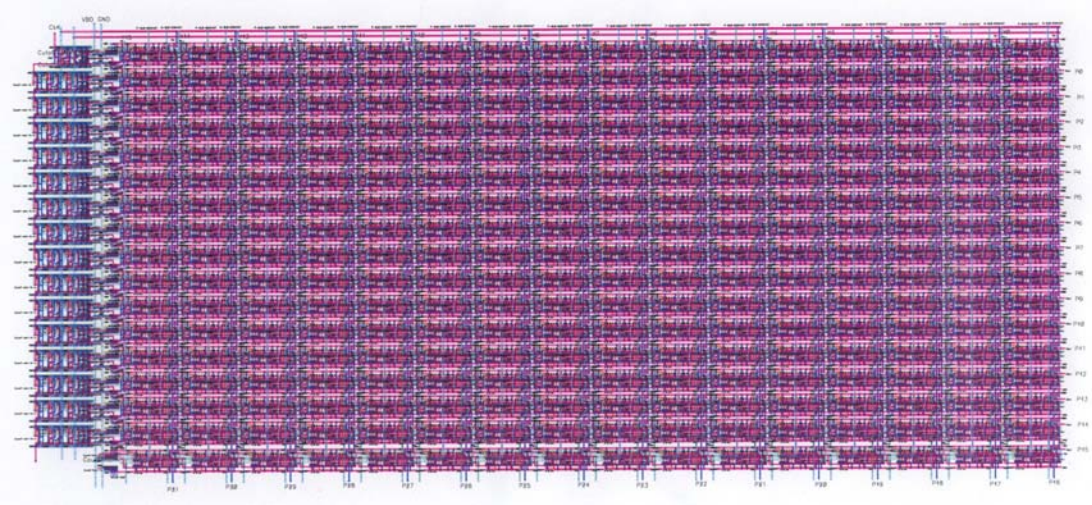


Figure 7.10: The Layout of the 16-bit Multiplier with DPCT

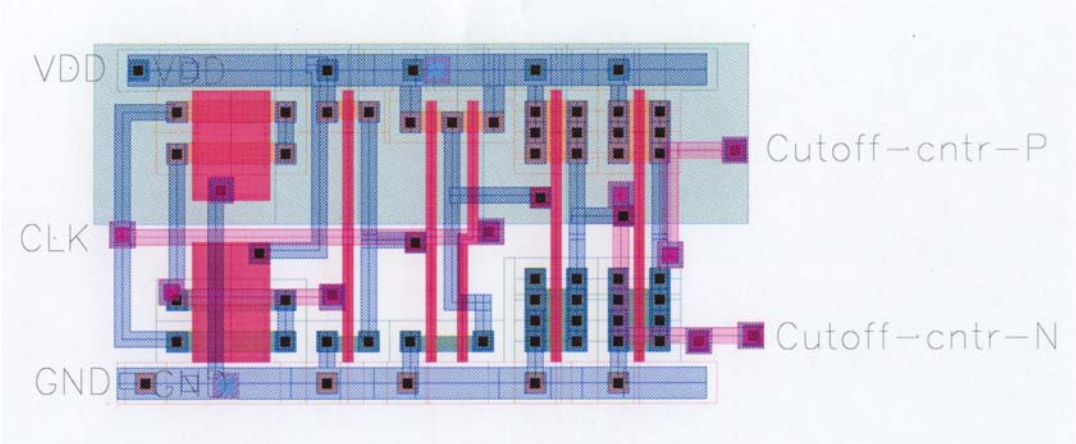


Figure 7.11: The Layout of the Cutoff Control Generator for Group 0

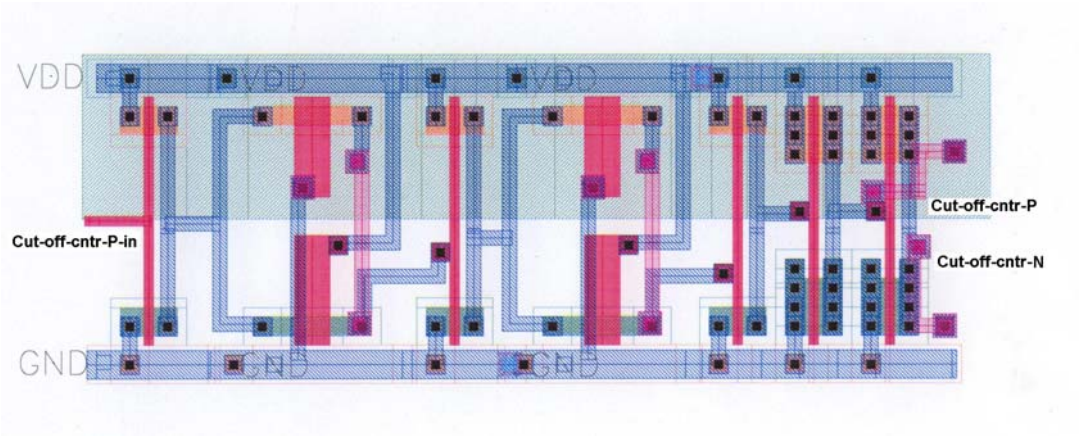


Figure 7.12: The Layout of the Cutoff Control Shifter

of the cutoff control signals for each group are all the same (120ps) corresponding to its previous group. So, we use one clock generator to generate the cutoff control signals for the first group: group 0. Then, we use a shifter, consisting of two C-switches and five $INVERTER$s, at each of the other groups from 1 to 15, to shift the cutoff signal from its previous group by 120ps to get the cutoff control signals for this group. The complete layout of the 16-bit multiplier with DPCT is shown in Figure 7.10. The cutoff transistors, the clock generator and shifters are on the left side of the layout. Figure 7.11 shows the cutoff control generator used for group 0. Figure 7.12 shows the shifter used for groups 1 to 15, which generates two cutoff control signals by shifting the input 120ps. Figure 7.13 shows the two cutoff transistors used for each group, whose sizes are 30$\times$ those of the minimal transistor.

7.3.1.3 Experimental Results for the 16-bit Multiplier with and without DPCT

After we obtained the layouts of the 16-bit multiplier with DPCT and without DPCT, we extracted the circuit netlists from the two layouts using the CADENCE layout extractor. Due to the limitations of the extractor, no inductances are extracted from the layout. Then, we simulated and verified both circuits using CADENCE SPECTRE^{TM}. Finally, we used Synopsys Nanosim^{TM} to get the power profiles of the two circuits. Table 7.1 shows the comparison of the two circuits.

The 16-bit multiplier with DPCT saves 54.7% of the total power, 85.7% of the active leakage power (including short-circuit power) and 38.1% of the dynamic power. The cost is 6.5%

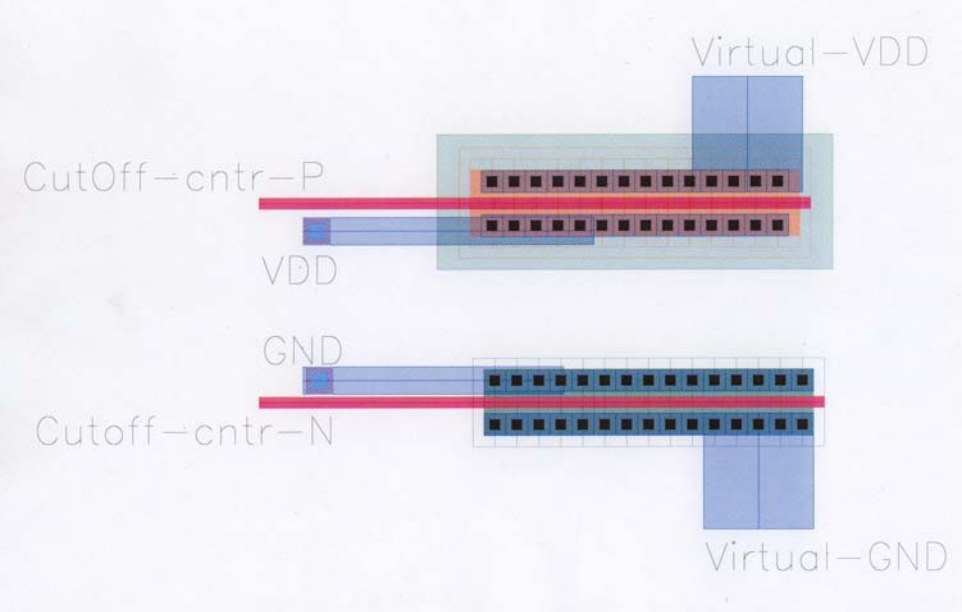


Figure 7.13: The Layout of the Cutoff MOSFETs for One Group

Table 7.1: Performance of DPCT on the Layout Design of a 16-bit Multiplier

	Total Power (μW)	Active Leakage Power (μW)	Dynamic Power (μW)	Delay (ps)	Area (μm^2)
No DPCT	1089.57	336.99	752.58	2976	6380
With DPCT	477.71	23.85	453.86	3170	6930
Difference	-54.73%	-85.71%	-38.12%	+6.52%	+8.62%

Table 7.2: Comparison of DPCT's Performance on c6288 and Layout Level 16-bit Multiplier

	Total Power Saving	Active Leakage Power Saving	Dynamic Power Saving	Delay Overhead	Area Overhead	Clock Rate (MHz)
Layout Design	54.7%	85.7%	38.1%	6.5%	8.6%	300
c6288	71.6%	86.6%	53.6%	6.0%	6.0%	200

delay overhead and 8.6% area overhead. In Chapter 6, we showed the power saving results of DPCT on *ISCAS '85* benchmarks, where c6288 is a 16-bit multiplier with an ideal transistor level netlist. So, we compared the power saving results of DPCT on c6288 and our layout design of the 16-bit multiplier. The results are shown in Table 7.2. The layout design gives a similar active leakage power saving. However, the dynamic power saving on the layout design is about 15% lower than c6288, which may be caused by the extra dynamic power dissipated on the parasitic capacitances in the layout design. The layout design also has a 33% higher clock rate than the transistor level c6288, which results in a higher percentage of dynamic power within the total power. (We use all minimal sized MOSFETs in c6288, but $2\times$ minimal sized or bigger MOSFETs in the layout design, which results in a higher clocking rate for the layout design.) Lower dynamic power savings and a higher dynamic power percentage in the layout design altogether result in about 15% less total power savings compared to the transistor level c6288. The area overhead of the layout design is a little higher than for the transistor level c6288, which is mainly caused by the extra routing space of the two sets of power lines, cut-off transistors and cutoff control generators. In spite of these differences, the results are still comparable.

7.4 Standard Cell Based Physical Design Using DPCT

Although hand-crafted custom-design can be used for physical design of ALUs, standard cell based designs are widely used for ASICs. To apply DPCT to those systems, the traditional standard cell based design flow has to be modified to implement DPCT automatically. We first show how to adapt the traditional standard cell based design flow to DPCT. Then, we show an example of the layout design of c432 using the modified standard cell based design flow.

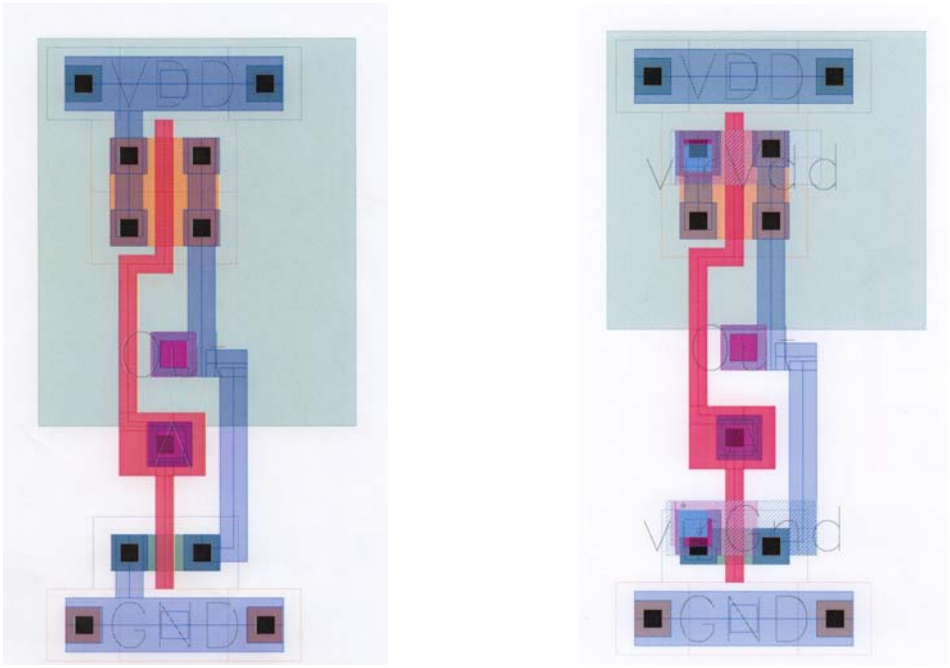
7.4.1 Adjustments of the Traditional Standard Cell Based Physical Design Flow for DPCT

The following are some possible adjustments to the traditional standard cell based design flow for DPCT.

7.4.1.1 Modification of the Physical Standard Cell Library

We already know that there are will be two sets of power lines in the DPCT standard cell layouts: the virtual *VDD*/*GND* and the global *VDD*/*GND*. Although all of the global *VDD*s and *GND*s will be connected together for all cells, only the virtual *VDD* and *GND* of those

cells that are in the same DPCT group are connected together. So, virtual *VDD* and *GND* have to be treated specially. They are more like another pair of ports to each gate, which should be routed according to the netlist.



(a) **(b)**

Figure 7.14: The *INVERTER* Layout of Traditional (a) and DPCT Standard Cell Library (b)

Figure 7.14 shows the *INVERTER* layout of the traditional and DPCT standard cell library. Both layouts have input *A*, output *Out* and *VDD*/*GND* lines. In the DPCT version layout, the *virVdd*/*virGnd* lines are the virtual power lines. *Metal*3 is used for routing the extra *virVdd* and *virGnd* lines. The advantage of using *Metal*3 is that the DPCT version standard cell layouts are almost the same size as the normal ones. If *Metal*1 was used, the DPCT cells will be around 30% bigger, which results in much bigger area overhead.

In addition to the changes to each available cell, some new cells have to added to the standard cell library, including the cutoff transistors and the cutoff control generators. Various sizes of cutoff transistor cells should be available. Various cutoff generators with the required delays should also be provided. These cells must be customized for each individual design.

7.4.1.2 Modification of the Logical Standard Cell Library

Two extra ports are added to each cell, which are the virtual *VDD* and *GND*. The logical standard cell library should also be modified accordingly. For example, the following shows

the Verilog modules of the standard inverter INV1 and the DPCT inverter INV2.

```
//Verilog modules of standard inverter INV1 and DPCT inverter INV2
module INV1 {A, Out};                    module INV2 {A, Out, virVDD, virGND};
   input A;                                  input A;
   output Out;                               output Out;
                                             inout virVDD, virGND;
endmodule                                endmodule
```

7.4.1.3 Modification of the Logic Synthesis and Layout Automatic Placement and Routing Tools

To generate the gate level netlist with DPCT automatically using the synthesis tools, the DPCT partitioning function should be added to these synthesis tools. To generate the layout with DPCT automatically, the automatic placement and routing tools should also be modified so that they can handle the routing of virtual *VDD*s and *GND*s.

7.4.2 Layout Design of c432 Using the Modified Standard Cell Based Design Flow

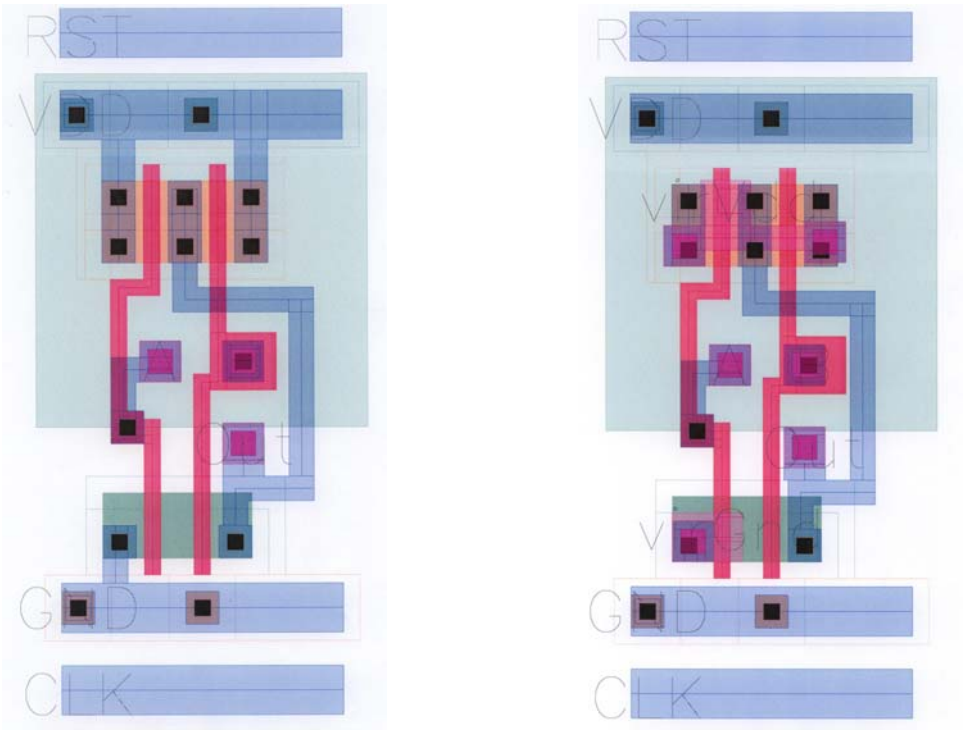
To verify the standard cell based design flow for DPCT, we implemented the layout of c432 with DPCT using a modified standard cell based design flow.

7.4.2.1 Steps of Standard Cell Based Layout Design of c432

The following are the steps for this design:

1. *Implement the Physical Standard Cell Library*

   We implemented two standard cell libraries: one for the normal circuits and one for DPCT circuits. Both of them include all types of gates used in the RUTMOD netlists of the *ISCAS '85* benchmark circuits including INVERTER, XOR, XNOR, NAND, NOR, AND and OR gates. The number of inputs for NAND, NOR, AND and OR gates ranges from 2 up to 9. The DPCT library cells are of the same size as the normal library cells, but with two extra pins: virtual *VDD* and virtual *GND*. Figure 7.15 shows the NAND2 gates of the normal and DPCT standard cell library. In the DPCT NAND2 layout, the *virVdd*/*virGnd* lines are the virtual power lines. The RST and CLK signals on both cells are dummy connections for the flip-flop cells used in the sequential circuits.



(a) **(b)**

Figure 7.15: The *NAND*2 Layout of Traditional (a) and DPCT Standard Cell Library (b)

2. *Implement the Logical Standard Cell Design Library*

   We also created a verilog file for each physical standard cell library, which defines the interfaces of all cells in each library. These files are required in the Cadence Silicon Ensemble automatic layout design tool. All the DPCT cells have two more pins: *virVdd* and *virGnd*, which are the virtual power lines.

3. *Partition the Circuit*

   To implement DPCT on c432, we use the heuristic partitioning algorithm we described in Chapter 4 to partition the circuit. Based on our previous layout implementation of the 16-bit multiplier, the layout implementations of DPCT tend to have higher area overhead and lower power savings than what we predict with the ideal logic level netlists. This is mainly due to the parasitic capacitances in the layout design. So, DPCT is more effective when there are fewer groups and more gates in each group so that the area overhead is minimized. Originally, our partitioning algorithm partitioned the circuit into 13 groups, with an average of 12 gates per group and a 35% estimated area overhead. We tried this partitioning in a layout implementation and got almost no power savings. So, we

adjusted the *pb* parameter in Equation 4.3 in the partitioning algorithm to get a 4-group partition, such that the average number of gates per group is 40, the minimal number of gates in a group is 21 and the estimated area overhead is 15%. This gives us 22.46% power savings due to the low area overhead.

4. *Generate the Gate Level Netlist*

   We implemented a C program to generate the verilog netlist with DPCT after the heuristic partitioning. The program also generated a non-DPCT verilog netlist as a reference. In both of the two netlists, we used the same types of gates as those in the original RUTMOD netlist. The power cutoff transistors are included in the DPCT netlist. But, the cutoff control generators are not included because they need full custom design. The cutoff control signals are treated as the primary inputs in the DPCT verilog netlist.

5. *Generate the Layout of c432 Using the Standard Cell Library*

   After we obtained the standard cell libraries and the gate-level c432 netlists for both DPCT and non-DPCT, we used Cadence $SiliconEnsemble^{TM}$ to generate the layout of c432 for both DPCT and non-DPCT versions. The non-DPCT layout is a complete layout. However, the cutoff control generators have to be added to the DPCT layout to make it complete. Figure 7.16 shows the final layout of c432 without DPCT.

6. *Add Cutoff Control Generators to the DPCT Layout*

   The cutoff control generators are manually designed based on the timing window generated by the partitioning algorithm. For this case, four cutoff control generators are needed, whose timing windows are $(0ps, 392ps)$, $(218ps, 657ps)$, $(501ps, 887ps)$ and $(795ps, 982ps)$, respectively. Here, two times the minimal switching window size is used. The number of gates in each group are 67, 49, 21 and 23, respectively. So, nearly $3/4$ of the gates are in groups 1 and 2. Since the duty cycles of the first three groups are close to 0.5 of the clock cycle, we can use the clock and shifted clock signals as their cutoff control signals. For the first group, the clock *CLK* and its inversion *CLK_BAR* are used as its *p*MOSFET and *n*MOSFET cutoff control signals, respectively, so that the gates in this group are turned on in the first half of the clock cycle. For the second group, both *CLK* and *CLK_BAR* are shifted by $218ps$ to be used as its cutoff control signals, so that the gates in this group are turned on during the timing window of $(218ps, 657ps)$. Figure 7.17 shows the layout of the shifter that shifts the clock signal by $218ps$. For the

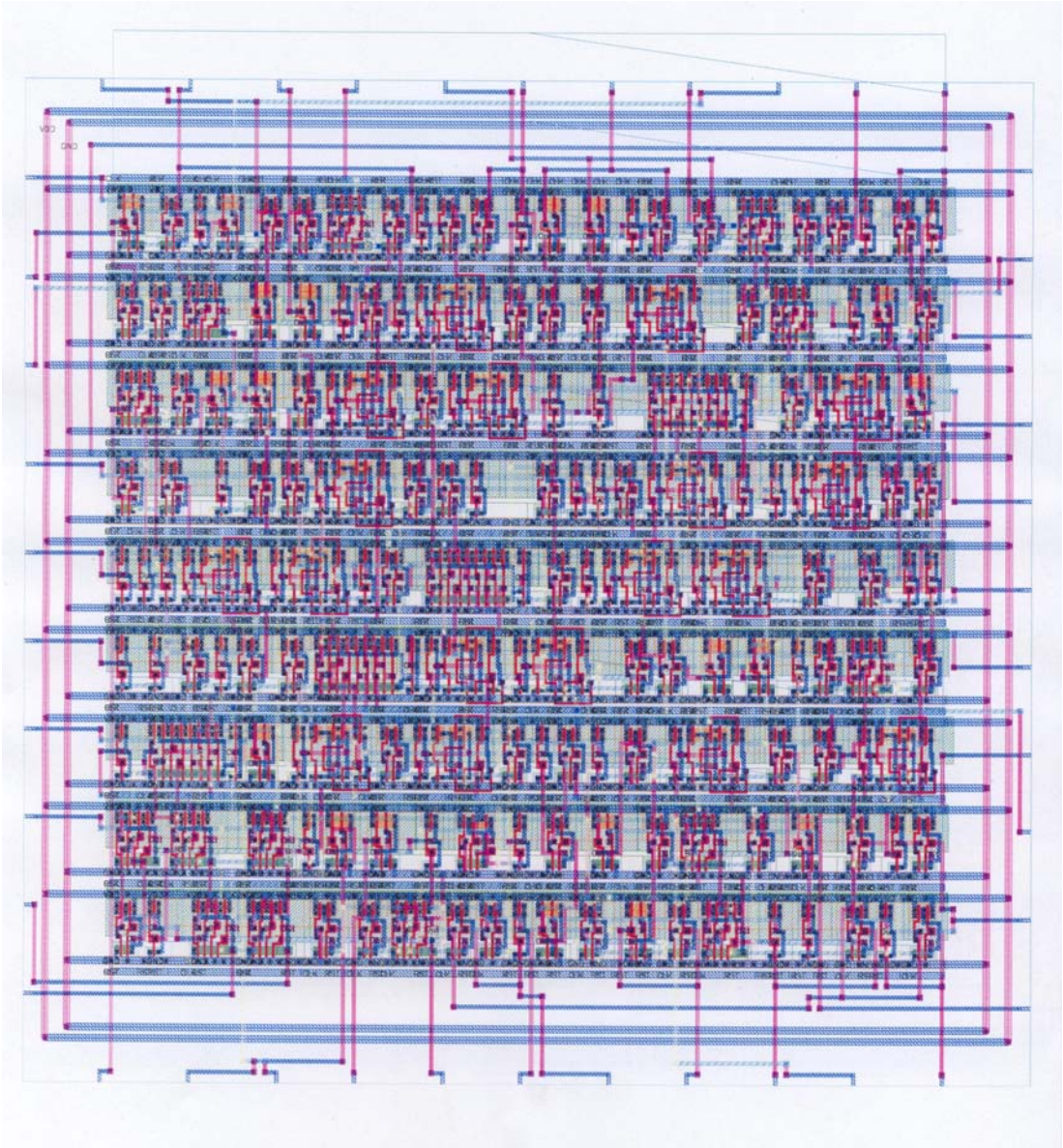


Figure 7.16: The Layout of c432 without DPCT

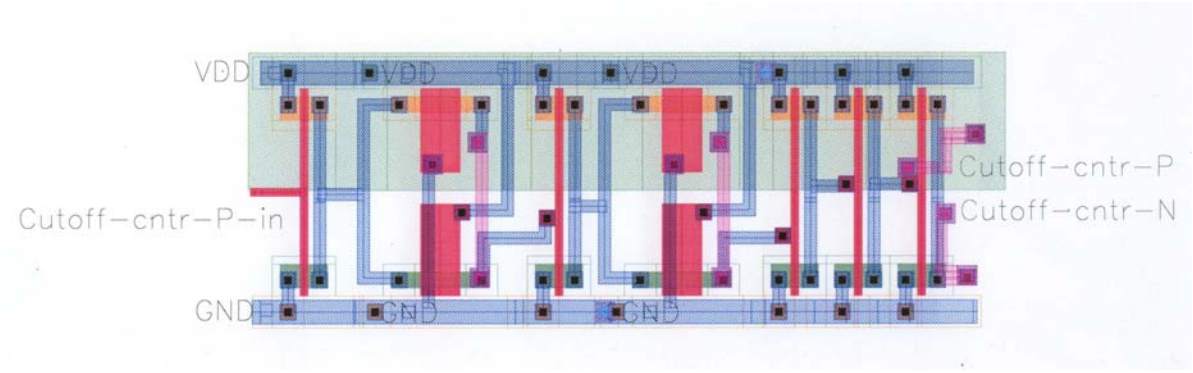


Figure 7.17: The Layout of the Cutoff Control Shifter Used in c432

third group, the *CLK_BAR* and *CLK* are used as its *p*MOSFET and *n*MOSFET cutoff control signals, respectively, so that the gates in this group are turned on in the second half of the clock cycle. As the number of gates in group 4 is small, we also use *CLK_BAR* and *CLK* as its *p*MOSFET and *n*MOSFET cutoff control signals, so that the gates in this group are also turned on in the second half of the clock cycle. These simplifications greatly reduce the area and dynamic power overhead of the DPCT design, whose benefit is more than the reduction of the leakage power saving. With these simplifications, we obtained 22.46% total power saving, 73.55% active leakage power saving and 2.29% dynamic power saving. If we add a dedicated cutoff control generator to group 4, we only obtained 20.76% total power saving, 73.98% active leakage power saving and 0.24% dynamic power increase. The extra reduction in the leakage power is less than the extra increase in the dynamic power. As a result, we got less total power saving and higher area cost by adding an extra cutoff control generator for group 4. So, using *CLK* and *CLK_BAR* as its cutoff control signals is a better design.

7.4.2.2 Experimental Results of the Standard Cell Based Layout Design of c432

After we created the layouts of c432 for DPCT and non-DPCT, we extracted the netlists from both the DPCT and non-DPCT layouts. We then used the same random input vectors with $1GHz$ frequency to test the two extracted circuits. The outputs of the DPCT circuit matched the outputs of the non-DPCT circuit. This verifies the construction of our c432 layout design with DPCT. We then used the Synopsys NanoSim simulator to get the detailed power profiles of the two circuits using the extracted netlists. The results are shown in Table 7.3. The c432 circuit

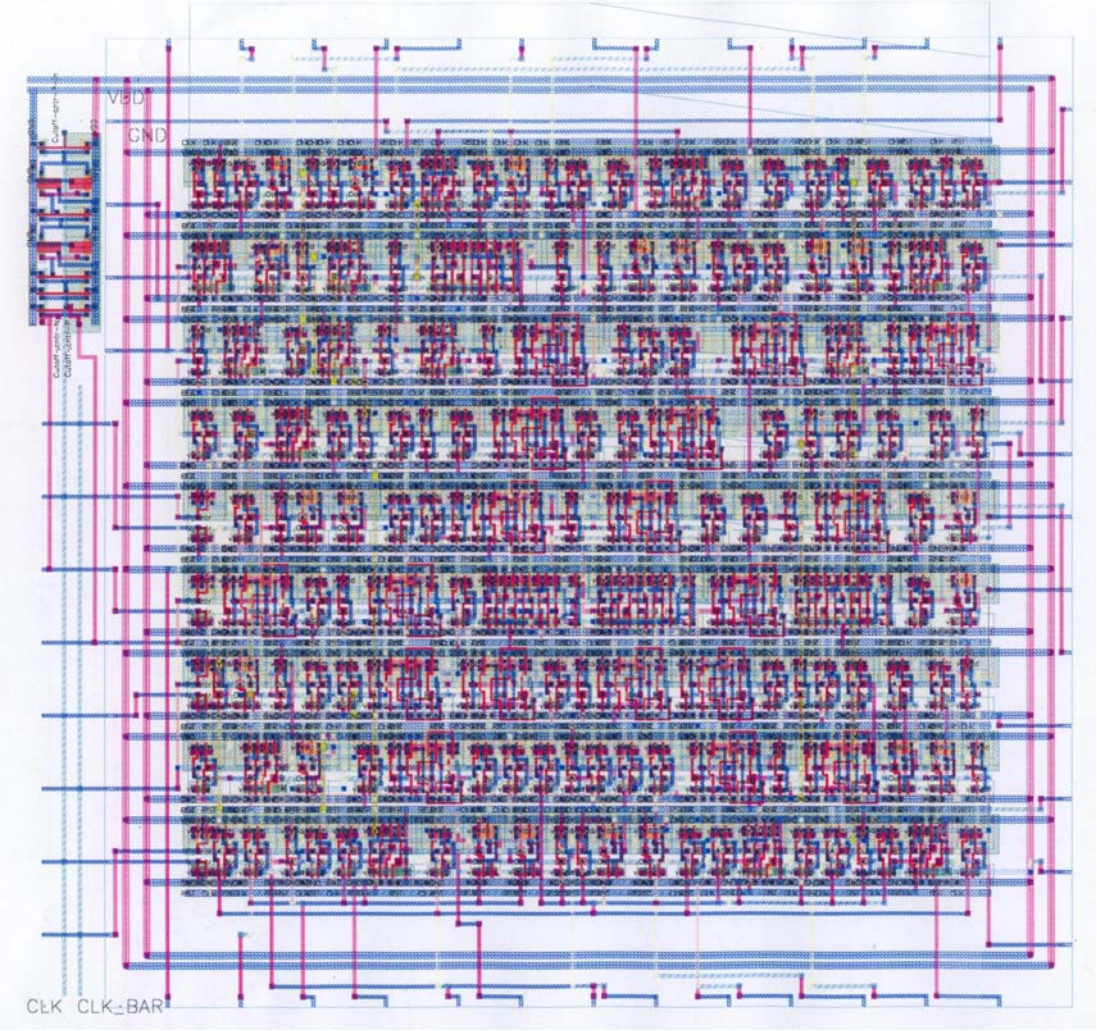


Figure 7.18: The Layout of c432 with DPCT

with DPCT saves 22.5% of the total power, 73.6% of the active leakage power (including short-circuit power) and 2.3% of the dynamic power, with 13.7% area overhead. The delay overhead is about 9%.

Table 7.3: Performance of DPCT on the Layout Design of c432

	Total Power (μW)	Active Leakage Power (μW)	Dynamic Power (μW)	Area (μm^2)
No DPCT	274.42	77.67	196.75	1296.00
With DPCT	212.78	20.54	192.24	1480.00
Difference	-22.46%	-73.55%	-2.29%	+13.70%

7.5 Does DPCT Support the Power Saving Mode with a Slowed Clock Rate?

After power of a DPCT cutoff group is turned off, the virtual VDD and GND of that group will collapse gradually. This will lead to the state loss for the gates within the group. The time that gates can hold their states depends on the following two factors: a) the capacitances of the virtual VDD and GND lines of that group; and b) the leakage current between the virtual VDD and virtual GND signals of that group. These two factors vary group by group in a circuit as they all depend heavily on the detailed circuit architecture.

We did an experiment to study how long it takes for the virtual VDD/GND signals to collapse using our layout implementation of c432, which has 4 DPCT groups. Initially, all power cutoff transistors are turned on, so that all virtual VDD/GND lines are fully charged to the real VDD/GND levels. Then, we turned off all of the power cutoff transistors and measured the voltage level changes of each virtual VDD/GND signal with the time to decay. In our experiments, VDD is $1V$. We use the 75% value of VDD as the lowest voltage for logic "1", which is $0.75V$. Similarly, we use $0.25V$ as the highest voltage for logic "0". When either virtual VDD drops below $0.75V$ or virtual GND rises above $0.25V$, the gates within this group start losing their states. Table 7.4 shows how long it takes for each virtual VDD signal to drop below $0.75V$ (at time t_1) and each virtual GND signal to rise above $0.25V$ (at time t_2). It also shows how long the gates in each DPCT group can hold their states, which is the minimum of t_1 and t_2 for that DPCT group.

We see that it is always the virtual VDD signal that first drops below $0.75V$. Also, Group 0 collapses more quickly than the other groups because most of the gates within Group 0 are

Table 7.4: Time for Virtual VDD/GND to Collapse in c432

DPCT Group	Time for virtual VDD to drop below $0.75V$ (ps)	Time for virtual GND to rise above $0.25V$ (ps)	Time for gates to lose states (ps)
Group 0	30	1015	30
Group 1	128	415	128
Group 2	185	1016	185
Group 3	349	881	349

inverters, which have much higher leakage current between virtual VDD and virtual GND than the other more complex CMOS gates. This is because the stacking effect in complex CMOS gates reduces the leakage current. As the time for gates to hold their states depends heavily on the circuit architecture, we cannot give a general guideline for this. It has to be studied case by case using analog simulations.

If the clock period has been extended in power saving mode, DPCT may fail if the virtual VDD/GND signals of those groups with primary outputs collapse before the start of next clock cycle. As the time for gates to hold their states depends heavily on the circuit architecture, we cannot give a general guideline for how much the clock can be slowed before DPCT fails. This has to be decided case by case using analog simulations.

One solution to avoid the failure of DPCT at a slowed clock rate is to add a latch to each primary output. The data can be latched before the power cutoff, so that the state can always be kept until the next clock cycle comes no matter how slow the clock is.

7.6 Summary

In this chapter, we presented the layout implementations of a 16-bit multiplier and c432 with DPCT. The 16-bit multiplier with DPCT saves 54.7% of the total power, 85.7% of the active leakage power (including short-circuit power) and 38.1% of the dynamic power with 7.7% delay overhead and 8.6% area overhead. The c432 circuit with DPCT saves 22.5% of the total power, 73.6% of the active leakage power (including short-circuit power) and 2.3% of the dynamic power with 9% delay overhead and 13.7% area overhead. The experimental results confirm the effectiveness of DPCT in physical level design. We also discussed some issues in the layout design of DPCT. Generally, DPCT can be easily adopted in those hand-crafted layout designs. However, some modifications have to be made to adapt DPCT to the standard cell based ASIC designs.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

We presented a novel low power design technique called DPCT that can reduce active leakage, standby leakage and dynamic power by applying the dynamic power cutoff technique to a circuit. We proposed a six-step approach to apply DPCT to a circuit automatically. The power savings and implementation costs of DPCT were presented and discussed. We also did power grid analysis and process variation analysis on DPCT. Experimental results on *ISCAS '85* benchmarks at the logic level modeled by 70*nm* Berkeley Predictive Models show up to 90% active leakage, 99% standby leakage, 54% dynamic power and 72% total power savings. DPCT can also reduce the maximal voltage drop on the power grid by more than 30% on average. With process variations, the average total power and active leakage power savings will be reduced by 12.7% and 14.8%, respectively. In spite of that, DPCT still gives excellent power savings, which are 73.6% of the active leakage power and 34.7% of the total power. We also implemented the layouts of a 16-bit multiplier and c432 using DPCT. The 16-bit multiplier with DPCT saves 54.7% of the total power, 85.7% of the active leakage power (including short-circuit power) and 38.1% of the dynamic power with 7.7% delay overhead and 8.6% area overhead. The c432 circuit with DPCT saves 22.5% of the total power, 73.6% of the active leakage power (including short-circuit power) and 2.3% of the dynamic power with 9% delay overhead and 13.7% area overhead. The experimental results on the layout designs confirmed the effectiveness of DPCT in physical level design.

8.2 Future Work

DPCT is a new low power technique. Although we did much theoretical research on it, much detailed design work has to be done to make it more easily and effectively usable in practical ASIC designs. The following are possible future work on DPCT.

1. *Adapt DPCT into the Standard Cell Based ASIC Design Flow*. As we discussed in Chapter 7, many modifications have to be made to adapt DPCT into the standard cell based ASIC design flow. Both the standard cell libraries and the CAD tools have to be modified to do this. There is still much work to be done in this area.

2. *Combine DPCT with Other Low Power Design Techniques*. In practical ASIC designs, many low power techniques are usually combined to achieve the best performance. So far, we only use DPCT independently. However, it is possible to combine DPCT with other low power design techniques, such as dual-V_{th} and dual V_{DD} techniques, to achieve better performance. More work can be done to verify the effectiveness of combining DPCT with other techniques.

3. *Verify DPCT Using Industry Benchmark Circuits*. In our research, we used the academic *ISCAS '85* benchmark circuits for our experiments. Although these academic benchmark circuits are good for research on new techniques, bigger industry benchmarks are better candidates for verifying new techniques. So, industry benchmark circuits, if available, may be used to further verify the performance of DPCT

4. *Verify DPCT by Fabricating Chips*. The best way to verify a new technique is to fabricate a chip. So, a chip can be fabricated to test the performance of DPCT.

Appendix A

User's Guide

A.1 Heuristic Partitioning of Circuits

The partitioning algorithm is implemented in a C program. It takes a circuit netlist in RUTMOD format as input. Then, it does the static timing analysis and heuristic partitioning. Finally, it generates the SPECTRE netlist of the partitioned circuit. All the circuits are modeled using the 70*nm* Berkeley Predictive Models.

- The path to the heuristic partitioning tool is given below:

  $/caip/u21/baozhen/research/leakage/DPCT/PSTRu2spectre/\ PSTRu2spectre\_dpct.sun4$

- To use this tool, type in as the following at the command line:

  $$PSTRu2spectre\_dpct.sun4 \quad file1 \quad file2$$

  where $file1$ is the input circuit netlist in RUTMOD format, and $file2$ is the output file giving the static timing analysis results of the circuits. Also, $file1$ should include the ".rutmod" extension and $file2$ can be with or without an extension. The SPECTRE netlists for DPCT and non-DPCT circuits are saved in $file1.scs$ and $file1\_dpct.scs$, respectively.

A.2 Power Grid Analysis

We use OCEAN and PERL scripts to run the SPECTRE simulations, collect the waveform data and analyze the data for the power grid analysis. All of the related files are in the following directory:

$$/caip/u21/baozhen/research/leakage/analogSim/PowerGridAnalysis$$

A.3 Process Variation Analysis

A C program was written to do SSTA on the circuits using the Monte Carlo method. All of the related files are in the following directory:

/caip/u21/baozhen/research/leakage/DPCT /StatisticalSTA

A.4 Standard Cell Based Layout Design

We implemented two standard cell libraries: one for the normal circuits and one for DPCT circuits. Both of them include all types of gates used in the RUTMOD netlists of the *ISCAS '85* benchmark circuits including the INVERTER, XOR, XNOR, NAND, NOR, AND and OR gates. The number of inputs for the NAND, NOR, AND and OR gates ranges from 2 up to 9. Both of the two libraries are designed using the 70*nm* Berkeley Predictive process. The paths to the two libraries are given below:

*/caip/u21/baozhen/cadence/mytechfiles/*70*nanometer_prim*

*/caip/u21/baozhen/cadence/mytechfiles/*70*nanometer_prim_dpct*

References

[1] International Technology Roadmap for Semiconductors, 1999. http://www.itrs.net/, SIA.

[2] R250 Pseudo-Random Number Generator. http://www.cs.sunysb.edu/algorith /implement/random-numbers/distrib/r250.seq.

[3] A. Abdollahi, F. Fallah, and M. Pedram. Runtime Mechanisms for Leakage Current Reduction in CMOS VLSI Circuits. In *Proc. of the Int'l. Symp. on Low Power Electronics and Design*, pages 213–218, Aug. 2002.

[4] A. Abdollahi, F. Fallah, and M. Pedram. Leakage Current Reduction in CMOS VLSI Circuits by Input Vector Control. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, 12(2):140–154, Feb. 2004.

[5] A. Agarwal, D. Blaauw, and V. Zolotov. Statistical Timing Analysis for Intra-Die Process Variations with Spatial Correlations. In *Proc. of the IEEE/ACM Int'l. Conf. on Computer-Aided Design (ICCAD)*, pages 900–907, 2003.

[6] A. Agarwal, D. Blaauw, V. Zolotov, S. Sundareswaran, M. Zhao, K. Gala, and R. Panda. Statistical Delay Computation Considering Spatial Correlations. In *Proc. of the Asia and South Pacific Conf. on Design Automation*, pages 271–276, 2003.

[7] A. Agarwal, S. Mukhopadhyay, A. Raychowdhury, K. Roy, and C. H. Kim. Leakage Power Analysis and Reduction for Nanoscale Circuits. *IEEE Micro*, 6(2):68–80, 2006.

[8] F. A. Aloul, S. Hassoun, K. A. Sakallah, and D. Blaauw. Robust SAT-based Search Algorithm for Leakage Power Reduction. In *Proc. of the Int'l. Workshop on Power and Timing Modeling, Optimization and Simulation*, pages 167–177, Sept. 2002.

[9] F. Assaderaghi, D. Sinitsky, S. A. Parke, J. Bokor, P. K. Ko, and C. Hu. A Dynamic Threshold-voltage MOSFET (DTMOS) for Ultra-low Voltage VLSI. *IEEE Trans. on Electron Devices*, 44(3):414–422, Mar. 1997.

[10] B. Yu and M. L. Bushnell. A Novel Dynamic Power Cutoff Technique (DPCT) for Active Leakage Reduction in Deep Submicron CMOS Circuits. In *Proc. of the Int'l. Symp. on Low Power Electronics and Design*, pages 214–219, Oct 2006.

[11] B. Yu and M. L. Bushnell. Power Grid Analysis of Dynamic Power Cutoff Technology. In *Proc. of the IEEE Int'l. Symp. on Circuits and Systems*, May 2007.

[12] S. Bhunia, N. Banerjee, Q. Chen, H. Mahmoodi, and K. Roy. A Novel Synthesis Approach for Active Leakage Power Reduction Using Dynamic Supply Gating. In *Proc. of the ACM/IEEE Design Automation Conf.*, pages 479–484, June 2005.

[13] S. Borkar. Design Challenges of Technology Scaling. *IEEE MICRO*, 19(4):23–29, July-Aug. 1999.

[14] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De. Parameter Variations and Impact on Circuits and Microarchitecture. In *Proc. of the IEEE/ACM Design Automation Conf.*, pages 338–342, 2003.

[15] D. Braha and O. Maimon. *A Mathematical Theory of Design: Foundations, Algorithms and Applications*. Kluwer, Boston, first edition, 1998.

[16] Y. Cao, Y. Lee, T. Chen, and C. C. Chen. HiPRIME: Hierarchical and Passivity Reserved Interconnect Macromodeling Engine for RLKC Power Delivery. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 24(6):797–806, June 2005.

[17] Y. Cao, T. Sato, M. Orshansky, D. Sylvester, and C. Hu. New Paradigm of Predictive MOSFET and Interconnect Modeling for Early Circuit Design. In *Proc. of the IEEE Custom Integrated Circuits Conf.*, pages 201–204, June 2000.

[18] H. Chang and S. S. Sapatnekar. Statistical Timing Analysis Considering Spatial Correlations using a Single Pert-Like Traversal. In *Proc. of the IEEE/ACM Int'l. Conf. on Computer-Aided Design (ICCAD)*, pages 621–625, 2003.

[19] T. Chen and C. C. Chen. Efficient Large-Scale Power Grid Analysis Based on Preconditioned Krylov-Subspace Iterative Methods. In *Proc. of the ACM/IEEE Design Automation Conf.*, pages 559–562, 2001.

[20] Z. Chen, C. Diaz, J. D. Plummer, M. Cao, and W. Greene. 0.18 μm Dual V_t MOSFET Process and Energy-Delay Measurement. In *Proc. of the 1996 Int'l. Electron Devices Meeting*, pages 851 – 854, Dec. 1996.

[21] Z. Chen, M. Johnson, L. Wei, and K. Roy. Estimation of Standby Leakage Power in CMOS Circuits Considering Accurate Modeling of Transistor Stacks. In *Proc. of the Int. Symp. on Low Power Electronics and Design*, pages 239–244, 1998.

[22] J. Choi, L. Wan, M. Swaminathan, B. Becker, and R. Master. Modeling of Realistic On-chip Power Grid using FDTD Method. In *Proc. of the Intl. Symp. on Eelectromagnetic Compatibility*, pages 238–243, 2002.

[23] V. De and S. Borkar. Technology and Design Challeges for Low Power and High Performance. In *Proc. of the Int'l. Symp. on Low-Power Electronics and Design*, pages 163–168, Aug. 1999.

[24] A. Devgan and C. Kashyap. Block-Based Static Timing Analysis with Uncertainty. In *Proc. of the IEEE/ACM Int'l. Conf. on Computer-Aided Design (ICCAD)*, pages 607–613, November 2003.

[25] D. Duarte, Y. F. Tsai, N. Vijaykrishnan, and M. J. Irwin. Evaluating Run-time Techniques for Leakage Power Reduction. In *Proc. of the 7th Asia and South Pacific and 15th Int. Conf. on VLSI Design*, pages 31–38, 2002.

[26] M. Eisele, J. Berthold, D. Schmitt-Landsiedel, and R. Mahnkopf. The Impact of Intra-die Device Parameter Variations on Path Delays and on the Design for Yield of Low Voltage Digital Circuits. In *Proc. of the Int'l. Symp. on Low Power Electronics and Design*, pages 237–242, 1996.

[27] P. Feldmann and F. Liu. Sparse And Efficient Reduced Order Modeling of Linear Subcircuits with Large Number of Terminals. In *Proc. of IEEE/ACM Int'l. Conf. on Computer Aided Design (ICCAD)*, pages 88–92, 2004.

[28] D. Fotty. *MOSFET Modeling with SPICE*. Englewood Cliffs, NJ: Prentice-Hall, 1997.

[29] J. Halter and F. Najm. A Gate-Level Leakage Power Reduction Method for Ultra Low Power CMOS Circuits. In *Proc. of the IEEE Custom Integrated Circuits Conf.*, pages 475–478, 1997.

[30] Fei Hu. *Process-Variation-Resistant Dynamic Power Optimization for VLSI Circuits*. PhD thesis, Department of Electrical and Computer Engineering, Auburn University, 2006.

[31] W. Hung, Y. Xie, N. Vijaykrishnan, M. Kandemir, M. J. Irwin, and Y. Tsai. Total Power Optimization through Simultaneously Multiple-V_{DD} Multiple-V_{th} Assignment and Device Sizing with Stack Forcing. In *Proc. of the Int'l. Symp. on Low Power Electronics and Design*, pages 144–149, 2004.

[32] J. Jacobs and D. Antoniadis. Channel Profile Engineering for MOSFET's with 100*nm* Channel Lengths. *IEEE Trans. on Electron Devices*, 42(5):870–875, May 1995.

[33] M. Johnson, D. Somasekhar, and K. Roy. Leakage Control with Efficient Use of Transistor Stacks in Single Threshold CMOS. In *Proc. of the ACM/IEEE Design Automation Conf.*, pages 442–445, June 1999.

[34] H. Mahmoodi-Meimand K. Roy, S. Mukhopadhyay. Leakage Current Mechanisms and Leakage Reduction Techniques in Deep-Submicrometer CMOS Circuits. *Proc. of the IEEE*, 91(2):305–327, Feb. 2003.

[35] T. Karnik, S. Borkar, and V. De. Sub-90*nm* Technologies: Challenges and Opportunities for CAD. In *Proc. of the IEEE/ACM Int'l. Conf. on Computer-Aided Design*, pages 203–206, 2002.

[36] T. Karnik, Y. Ye, J. Tschanz, L. Wei, S. Burns, V. Govindarajulu, V. De, and S. Borkar. Total Power Optimization By Simultaneous Dual-V_t Allocation and Device Sizing in High Performance Microprocessors. In *Proc. of the ACM/IEEE Design Automation Conf.*, pages 486–491, June 2002.

[37] H. Kawaguchi, K. Nose, and T. Sakurai. A Super Cut-off CMOS (SCCMOS) Scheme for 0.5-V Supply Voltage with Picoampere Stand-By Current. *IEEE J. of Solid-State Circuits*, 35(10):1498–1501, October 2000.

[38] A. Keshavarzi, C. F. Hawkins, K. Roy, and V. De. Effectiveness of Reverse Body Bias for Low Power CMOS Circuits. In *Proc. of the 8th NASA Symp. on VLSI Design*, pages 231–239, 1999.

[39] M. Ketkar and S. S. Sapatnekar. Standby Power Optimization via Transistor Sizing and Dual Threshold Voltage Assignment. In *Proc. of the IEEE/ACM Int'l. Conf. on Computer-Aided Design (ICCAD)*, pages 375–378, 2002.

[40] C. H. Kim and K. Roy. Dynamic V_{th} Scaling Scheme for Active Leakage Power Reduction. In *Proc. of the Conf. on Design, Automation and Test in Europe*, pages 163–167, 2002.

[41] S. Kirkpatrick and E. P. Stoll. A Very Fast Shift-Register Sequence Random Number Generator. *IEEE J. of Computational Physics*, 40:517–526, 1981.

[42] J. N. Kozhaya, S. R. Nassif, and F. N. Najm. Multigrid-Like Technique for Power Grid Analysis. In *Proc. of the IEEE/ACM Int'l. Conf. on Computer-Aided Design (ICCAD)*, pages 480–487, 2001.

[43] T. Kuroda, T. Fujita, S. Mita, T. Nagamatsu, S. Yoshioka, K. Suzuki, F. Sano, M. Norishima, M. Murota, M. Kako, M. Kinugawa, M. Kakumu, and T. Sakurai. A $0.9V$ $150MHz$ $10mW$ $4mm$ 2-D Discrete Cosine Transform Core Processor with Variable Threshold Voltage Scheme. *IEEE J. of Solid-State Circuits*, 31:1770–1779, Nov. 1996.

[44] P. Li. Power Grid Simulation Via Efficient Sampling-Based Sensitivity Analysis And Hierarchical Symbolic Relaxation. In *Proc. of the ACM/IEEE Design Automation Conf.*, pages 664–669, 2005.

[45] J. Liou, A. Krstic, L. Wang, and K. Cheng. False-Path-Aware Statistical Timing Analysis and Efficient Path Selection for Delay Testing and Timing Validation. In *Proc. of the IEEE/ACM Design Automation Conf.*, pages 566–569, 2002.

[46] W. Liu and *et al.* BSIM3v3.2.1 MOSFET MODEL USERS' MANUAL. http://www.eecs.berkeley.edu/Pubs/TechRpts/1999/ERL-99-19.pdf, Mar. 1999.

[47] Y. Lu. *Power and Performance Optimization of Static CMOS Circuits with Process Variation*. PhD thesis, Department of Electrical and Computer Engineering, Auburn University, 2007.

[48] Y. Lu and V. D. Agrawal. Leakage and Dynamic Glitch Power Minimization Using Integer Linear Programming for V_{th} Assignment and Path Balancing. In *Proc. of the Power and Timing Modeling, Optimization and Simulation Workshop (PATMOS'05)*, pages 217–226, September 2005.

[49] Y. Lu and V. D. Agrawal. CMOS Leakage and Glitch Minimization for Power-Performance Tradeoff. *Journal of Low Power Electronics*, 2(3):378–387, December 2006.

[50] Y. Lu and V. D. Agrawal. Statictical Leakage and Timing Optimization for Submicron Process Variation. In *Proc. of the 20th International Conf. on VLSI Design*, pages 439–444, January 2007.

[51] L.Wei, Z. Chen, M. Johnson, K. Roy, Y. Ye, and V. De. Design and Optimization of Dual Threshold Circuits for Low Voltage Low Power Applications. *IEEE Trans. on VLSI Systems*, 7(1):16–24, March 1999.

[52] H. Mangassarian and M. Anis. On Statistical Timing Analysis with Inter- and Intra-die Variations. In *Proc. the Design, Automation and Test Conf. in Europe (DATE)*, pages 1530–1591, 2005.

[53] K. Min, H. Kawaguchi, and T. Sakurai. Zigzag Super Cut-off CMOS (ZSCCMOS) Block Activation with Self-Adaptive Voltage Level Controller: An Alternative to Clock-gating Scheme in a Leakage Dominant Era. *in Digest of Technical Papers of the IEEE Solid-State Circuits Conf.*, pages 400–502, 2003.

[54] S. Mutoh, T. Douskei, Y. Matsuya, T. Aoki, S. Shigematsu, and J. Yamada. 1-V Power Supply High-Speed Digital Circuit Technology with Multi-Threshold Voltage CMOS. *IEEE J. of Solid-State Circuits*, 30:847–854, Aug. 1995.

[55] S. R. Nassif. Modeling and Analysis of Manufacturing Variations. In *Proc. of the Custom Integrated Circuits Conf.*, pages 223–228, 2001.

[56] D. Nguyen, A. Davare, M. Orshansky, D. Chinnery, B. Thompson, and K. Keutzer. Minimization of Dynamic and Static Power Through Joint Assignment of Threshold Voltages and Sizing Optimization. In *Proc. of the Int'l. Symp. on Low Power Electronics and Design*, pages 158–163, 2003.

[57] M. Orshansky and A. Bandyopadhyay. Fast Statistical Timing Analysis Handling Arbitrary Delay Correlations. In *Proc. of the IEEE/ACM Design Automation Conf.*, pages 337–342, 2004.

[58] M. Orshansky and K. Keutzer. A General Probabilistic Framework for Worst Case Timing Analysis. In *Proc. of the IEEE/ACM Design Automation Conf.*, pages 556–561, 2002.

[59] P. Pant, V. K. De, and A. Chatterjee. Simultaneous Power Supply, Threshold Voltage and Transistor Size Optimization for Low-Power Operation of CMOS Circuits. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, 6(4):538 – 545, Dec. 1998.

[60] P. Pant, R. K. Roy, and A. Chattejee. Dual-Threshold Voltage Assignment with Transistor Sizing for Low Power CMOS Circuits. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, 9(2):390–394, Apr. 2001.

[61] R. Pierret. *Semiconductor Device Fundamentals*. Addison-Wesley, 1996.

[62] H. Qian, S. R. Nassif, and S. S. Sapatnekar. Random Walks in A Supply Network. In *Proc. of the ACM/IEEE Design Automation Conf.*, pages 93–98, 2003.

[63] T. Raja. A Reduced Constraint Set Linear Program for Low-Power Design of Digital Circuits. Master's thesis, Department of Electrical and Computer Engineering, Rutgers University, 2002.

[64] T. Raja. *Minimum Dynamic Power CMOS Design with Variable Input Delay Logic*. PhD thesis, Department of Electrical and Computer Engineering, Rutgers University, 2004.

[65] T. Raja, V. D. Agrawal, and M. L. Bushnell. Minimum Dynamic Power CMOS Circuit Design by a Reduced Constraint Set Linear Program. In *Proc. of the 16th Int'l. Conf. on VLSI Design*, pages 527–532, Jan. 2003.

[66] T. Raja, V. D. Agrawal, and M. L. Bushnell. CMOS Circuit Design for Minimum Dynamic Power and Highest Speed. In *Proc. of the 17th Int'l. Conf. on VLSI Design*, pages 1035 – 1040, Jan. 2004.

[67] T. Raja, V. D. Agrawal, and M. L. Bushnell. Variable Input Delay CMOS Logic Design for Low Dynamic Power Circuits. In *Proc. of the Power and Timing Modeling, Optimization and Simulation Workshop (PATMOS'05)*, pages 436–445, September 2005.

[68] T. Raja, V. D. Agrawal, and M. L. Bushnell. Transistor Sizing of Logic Gates to Maximize Input Delay Variability. *Journal of Low Power Electronics*, 2(1):121–128, April 2006.

[69] K. Roy and S. C. Prasad. *Low-Power CMOS VLSI Circuit Design*. New York: Wiley, 2000.

[70] K. Schuegraf and C. Hu. Hole Injection SiO2 Breakdown Model for Very Low Voltage Lifetime Extrapolation. *IEEE Trans. on Electron Devices*, 41(5):761–767, May 1994.

[71] G. Sery, S. Borkar, and V. De. Life Is CMOS: Why Chase the Life After? In *Proc. of the ACM/IEEE Design Automation Conf.*, pages 78–83, 2002.

[72] 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. In *Proc. of the ACM/IEEE Design Automation Conf.*, pages 436–441, 1999.

[73] H. Su, E. Acar, and S. R. Nassif. Power Grid Reduction Based on Algebraic Multigrid Principles. In *Proc. of the ACM/IEEE Design Automation Conf.*, pages 109–112, 2003.

[74] V. Sundararajan and K. K. Parhi. Low Power Synthesis of Dual Threshold Voltage CMOS VLSI Circuits. In *Proc. of the Int'l. Symp. on Low Power Electronics and Design*, pages 139–144, 1999.

[75] Y. Taur, A. Chandrakasan, W. J. Bowhill, and F. Fox. CMOS Scaling and Issues in Sub-0.25μm Systems. In *Proc. of IEEE Design of High-Performance Microprocessor Circuits*, pages 27–45, 2001.

[76] Y. Taur and T. H. Ning. *Fundamentals of Modern VLSI Devices*. Cambridge, United Kingdom, Cambridge Univ. Press, 1998.

[77] S. Thompson, P. Packan, and M. Bohr. Linear versus Saturated Drive Current: Tradeoffs in Super Steep Retrograde Well Engineering. *in Digest of Technical Papers of IEEE Symp. on VLSI Technology*, pages 154–155, 1996.

[78] J. W. Tschanz, S. G. Narendra, Y. Ye, B. A. Bloechel, S. Borkar, and V. De. Dynamic Sleep Transistor and Body Bias for Active Leakage Power Control of Microprocessors. *IEEE J. of Solid-State Circuits*, 38(11):1838–1845, 2003.

[79] S. Uppalapati, M. L. Bushnell, and V. D. Agrawal. Glitch-Free Design of Low Power ASICs Using Customized Resistive Feedthrough Cells. In *Proc. of the 9th VLSI Design & Test Symp. (VDAT'05)*, pages 41–49, August 2005.

[80] Siri Uppalapati. Low Power Design of Standard Cell Digital VLSI Circuits. Master's thesis, Department of Electrical and Computer Engineering, Rutgers University, 2004.

[81] H. J. M. Veendrick. Short-Circuit Dissipation of Static CMOS Circuitry and Its Impact on the Design of Buffer Circuits. *IEEE J. of Solid-State Circuits*, 19(4):468–473, Aug. 1984.

[82] S. Venkatesan, J. W. Lutze, C. Lage, and W. J. Taylor. Device Drive Current Degradation Observed with Retrograde Channel Profiles. In *Proc. of Int'l. Electron Devices Meeting*, pages 419–422, 1995.

[83] C. Visweswariah, K. Ravindran, K. Kalafala, S. G. Walker, and S. Narayan. First-order Incremental Block-Based Statistical Timing Analysis. In *Proc. of the ACM/IEEE Design Automation Conf.*, pages 331–336, 2004.

[84] Q. Wang and S. B. K. Vrudhula. Static Power Optimization of Deep Submicron CMOS Circuits for Dual V_t Technology. *Digest of Technical Papers of the IEEE/ACM Int'l. Conf. on Computer-Aided Design (ICCAD)*, pages 490 – 496, Nov. 1998.

[85] C. Wann, F. Assaderaghi, R. Dennard, C. Hu, G. Shahidi, and Y. Taur. Channel Profile Optimization And Device Design for Low-Power High-Performance Dynamic-Threshold MOSFET. *in Digest of Technical Papers of IEEE Int. Electron Devices Meeting*, pages 113–116, 1996.

[86] L. Wei, Z. Chen, M. Johnson, K. Roy, and V. De. Design and Optimization of Low Voltage High Performance Dual Threshold CMOS Circuits. In *Proc. of the IEEE/ACM Design Automation Conf.*, pages 489–494, June 1998.

[87] L. Wei, Z. Chen, K. Roy, Y. Ye, and V. De. Mixed-V_{th} (MVT) CMOS Circuit Design Methodology for Low Power Applications. In *Proc. of the ACM/IEEE Design Automation Conf.*, pages 430–435, June 1999.

[88] L. Wei, K. Roy, and C. Koh. Power Minimization by Simulataneous Dual-V_{th} Assignment and Gate-Sizing. In *Proc. of the IEEE Custom Integrated Circuit Conf.*, pages 413–416, 2000.

[89] N. H. E. Weste and D. Harris. *CMOS VLSI Design: A Circuits and Systems Perspective*. Pearson Education/Addison-Wesley, Boston, 2005.

[90] Y. Ye, S. Borkar, and V. De. A New Technique for Standby Leakage Reduction in High Performance Circuits. In *Proc. of the IEEE Symp. on VLSI Circuits*, pages 40–41, 1998.

[91] W. Yeh and J. Chou. Optimum Halo Structure for Sub-0.1μm CMOSFET's. *IEEE Trans. on Electron Devices*, 48(10):2357–2362, October 2001.

[92] Z. Zhu, B. Yao, and C. Cheng. Power Network Analysis Using an Adaptive Algebraic Multigrid Approach. In *Proc. of the ACM/IEEE Design Automation Conf.*, pages 105–108, 2003.

Curriculum Vita

Baozhen Yu

1990-94	B.S. in Electrical Engineering Nankai University, Tianjin, China
1994-97	M.S. in Electrical Engineering Nankai University, Tianjin, China
2002-07	Ph.D., in Electrical and Computer Engineering Rutgers University, New Brunswick, New Jersey

1997-99	Hardware Design Engineer Zhongxing Telecom Ltd., Shenzhen, China
1999-02	Senior System Design Engineer Cellon Ltd., Beijing, China
2002-06	Teaching Assistant, ECE Dept. Rutgers University, New Brunswick, New Jersey
2006-07	Research Assistant, WinLab, Rutgers University, New Brunswick, New Jersey

Oct, 2006	B. Yu and M. L. Bushnell, “A Novel Dynamic Power Cutoff Technique (DPCT) for Active Leakage Reduction in Deep Submicron CMOS Circuits,” in *Proc. of the Int’l. Symp. on Low Power Electronics and Design*, pages 214-219.
May, 2007	B. Yu and M. L. Bushnell, “Power Grid Analysis of Dynamic Power Cutoff Technology,” in *Proc. of the IEEE Int’l. Symp. on Circuits and Systems*, pages 1393-1396.