M.S. THESIS

Timing Error Aware Supply Voltage Control in Synchronous Circuits

동기 회로에서 시간 오류를 고려한 공급전압 제어

BY

Pierre NICOLAS-NICOLAZ

FEBRUARY 2015

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Timing Error Aware Supply Voltage Control in Synchronous Circuits

동기 회로에서 시간 오류를 고려한 공급전압 제어

지도교수 최 기 영

이 논문을 공학석사 학위논문으로 제출함

2014 년 11 월

서울대학교 대학원

전기컴퓨터 공학부

Pierre NICOLAS-NICOLAZ

Pierre NICOLAS-NICOLAZ 의 공학석사
학위논문을 인준함

2014 년 11 월

위 원 장 : 채수익_____

부위원장 : 최기영_____

위 원 : 김태환_____

Abstract

Timing Error Aware Supply Voltage Control in Synchronous Circuits

Pierre Nicolaz-Nicolaz

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

COLLEGE OF ENGINEERING

SEOUL NATIONAL UNIVERSITY

Modern embedded systems are becoming more and more constrained by power consumption. While we require those systems to compute even more data at faster speed, lowering energy consumption is essential to preserve battery life as well as integrity of devices.

Amongst many techniques to reduce power consumption of chips such as power gating, clock gating, etc., lowering the supply voltage (maybe reducing chip's frequency) is known to be the most effective one. However, lowering the supply voltage of chips too much down to near the threshold voltage of transistors causes the logic delay to vary exponentially with intrinsic and extrinsic variations (process variations, temperature, aging, etc.) and thus forces the designer to set increased

timing margin.

This thesis proposes a technique for automatically adjusting the supply voltage to match the speed of a logic block with a given time constraint. Depending on process and temperature variations, our technique chooses the minimum supply voltage to satisfy the timing constraint defined by the designer. This allows him/her to reduce the default supply voltage of the logic block and thus save power. In our experiments at the 28/32nm technology node, we succeeded in reducing the logic block power by 52% on average by varying the supply voltage between 0.55V and 1V, while the nominal supply voltage is 1.05V.

Keywords: Low-power, VLSI, Process variation, Near-Threshold-Voltage, Current-Completion Sensing-Device

Student Number: 2013-22514

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Constant decrease in transistor size, advancement in CAD tools and increased exigencies in chips has led to an exponentially increase of the number of transistors in a single device. From the thousands of transistors in an Intel 8080 chip in the 70's, we succeeded at cramming more than several billions of transistors in modern chips (for example, the IBM Power 7 with 80MB of L3 cache contains more than 2,100,000,000 transistors). This increase in processing power came at the cost an ever increase in power consumption. While several techniques have been developed to reduce power consumption (for example power gating), the main design parameter that chip manufacturers used in the past to improve power performance was reducing the nominal supply voltage. As we can see in Figure 1.1, from the 0.25µm technology node to the 45nm node, the supply voltage was reduced by a factor of 2.7. But since

the 45nm node, the supply voltage has stagnated around 1V and thus total energy of chips can difficultly be lowered down.

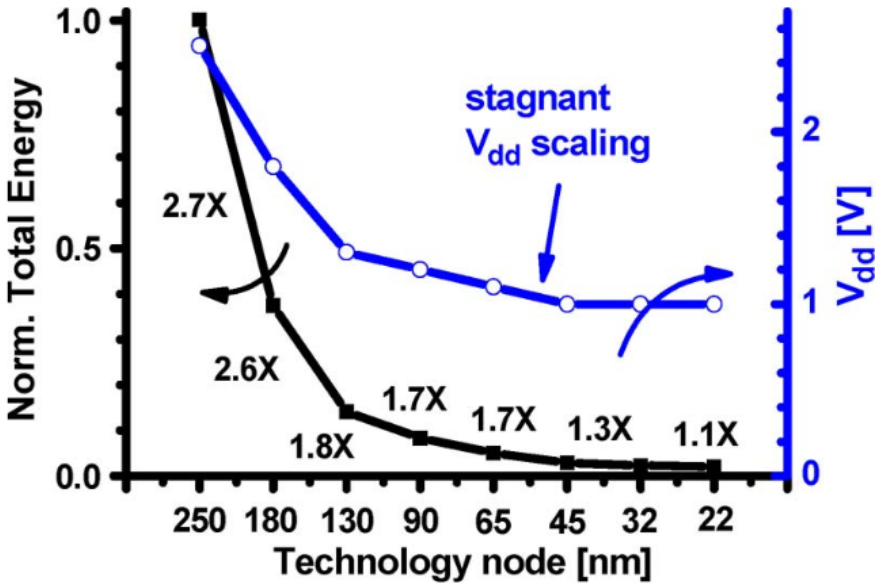


Figure 1.1 Technology scaling trends of supply voltage and energy. (ref. [1]).

To overcome this difficulty, it has been proposed to reduce the supply voltage of chips below their nominal values. While it is possible to use a supply voltage which even goes below the threshold voltage of a transistor (see the original work for example [2]), the speed of the transistor decreases exponentially and its sensibility to intrinsic and extrinsic variations is severely increased (variation of oxide thickness, doping, transistor gate width, temperature, supply voltage variations, aging, …). Furthermore, the cost of leakage energy due to decreased transistor speed overcome

the savings in dynamic energy at very low supply voltage, which push designers to design in the *Near-Threshold Voltage* region, thus making *Near-Threshold Computing*. Still, increased sensibilities to variations causes designers to use excessive timing margin and cause huge performance losses (see for example [3]).

We devised a technique to automatically adjust the supply voltage of a logic block to match a certain timing, despite the excessive variation. Our technique determines the correct voltage for the circuit to run at a given frequency. In traditional approach, the designer would increase supply voltage to match its timing constraint. Our approach allows the designer to make its circuit to match the timing constraint without considering variations, and adjusts the supply voltage at runtime. It allows power savings compared to the approach where the designer would just increase the supply voltage to match his constraint.

For that purpose, we used a Current-Sensing Completion Detecting (CSCD) circuit, which allows us to detect at each new clock cycle if the previous computation was finished. If an error is detected, the supply voltage of the chip is raised a little to prevent similar errors. If in the next cycles a new error is detected, we raise the supply voltage again and so on until no error is detected. This allows the circuit to run without timing errors once the right supply voltage is found. In our experiments at the 28/32nm technology node, it allowed us to reduce supply voltage by 52% on average by varying

the supply voltage of the logic between 0.55V and 1V.

The second chapter describes more detailed information about Near-Threshold Computing and the behavior of the CSCD device we used. Chapter 3 describes our approach and chapter 4 explains our experimental setup. Finally, we will present our results in chapter 6 and conclusion with remaining work for further improvement in chapter 7.

Chapter 2

Background

We will give here detailed information on which our work is based to allow readers to further understand the present document.

2.1 Near Threshold Computing

As explained before, a key technique to reduce power consumption in chips is to reduce the supply voltage since:

$$E_T = E_{DYN} + E_L = \alpha C_{load} V_{dd}^{\ 2} + I_{leak} T_D V_{dd} \qquad (2.1)$$

Where α is the switching probability of the transistors, C_{load} the load capacitance, I_{leak} constant associated with the technology used and T_D the leakage time. By reducing the supply voltage, dynamic energy is reduced according to this square law allowing quadratic power savings.

However, when the supply voltage is reduced under its nominal value, the transistors become slower: this increases leakage energy. Thus, supply voltage cannot

be reduced indefinitely. Instead, it exists an energy minimum point which balance the savings in dynamic energy and leakage energy (Figure 2.1).

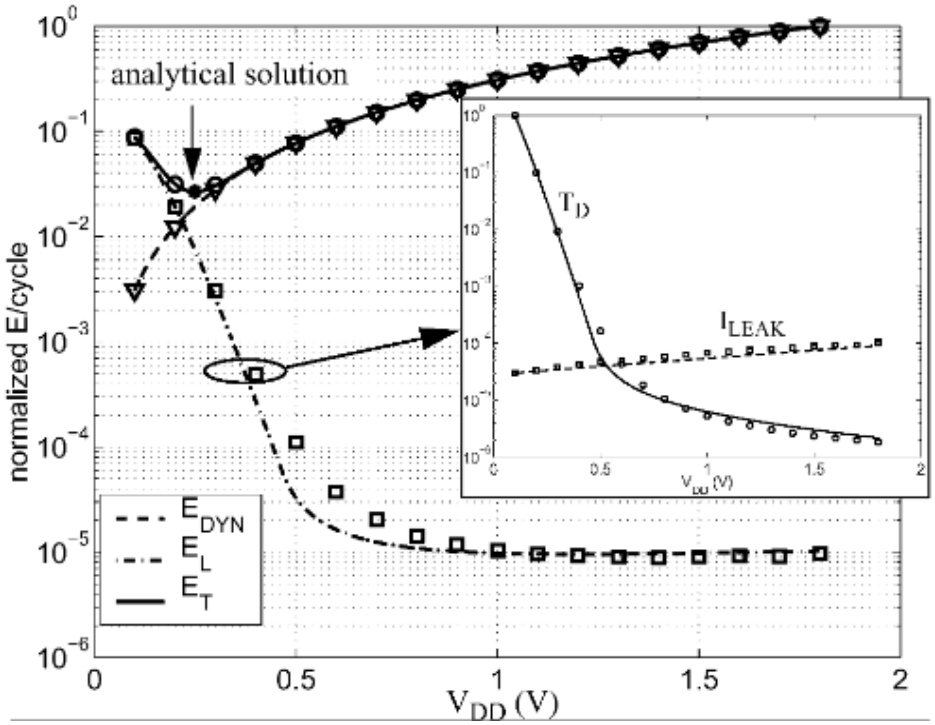


Figure 2.1. Model versus simulation of FIR filter showing minimum energy point and contribution of dynamic and leakage energy. Inset shows I_{LEAK} and T_D effects on E_L (ref. [4]).

Furthermore, other considerations at the architectural level are often advanced as a reason to increase the supply voltage above the threshold voltage of the transistors.

A common technique to regain the speed lost due to Near Threshold Computing is to use cluster of NTC cores ([5]). Different parameters are needed to find the optimum energy point, such as a synergy between the speed of the caches and the cores ([6]) or the amount of the application that can be made parallelizable ([7]).

2.2 Current Sensing Completion Detection

The current sensing completion detection was first described by Horowitz in 1994 as a way to efficiently replace complex handshaking protocols in self-timing (asynchronous) logic ([8]). The idea is that “when CMOS devices transition their outputs because of input variations, current flow dramatically increases. When all the inputs and outputs of a CMOS logic block reach their final logic state, the current flow decreases virtually to zero.” By monitoring the current, we can know if a CMOS device has finished its computation or not.

Although the original circuitry was complex and based on a BJT, the idea has been analyzed several times and is included in most references about asynchronous logics. Several papers tried to improve the circuit (such as [9]), but the simple idea behind the circuit is represented in Figure 2.2.

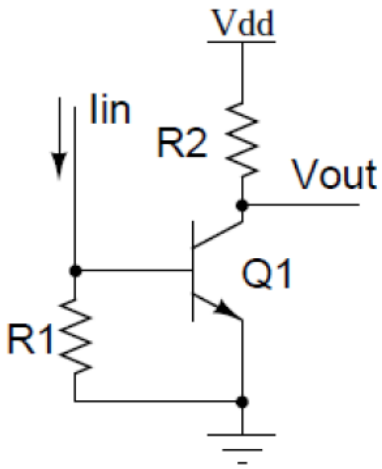
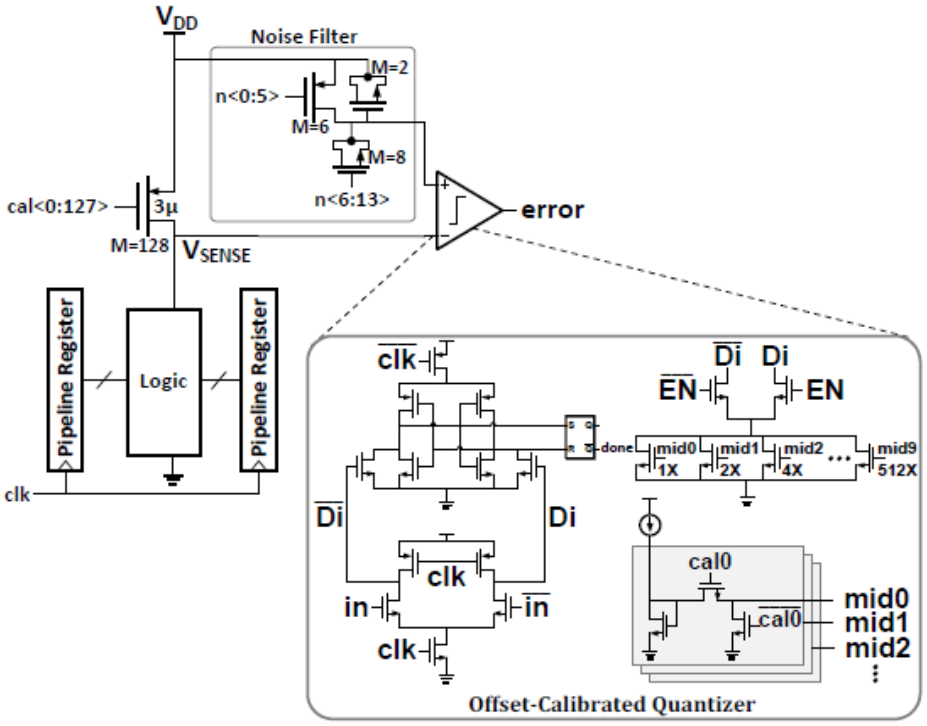


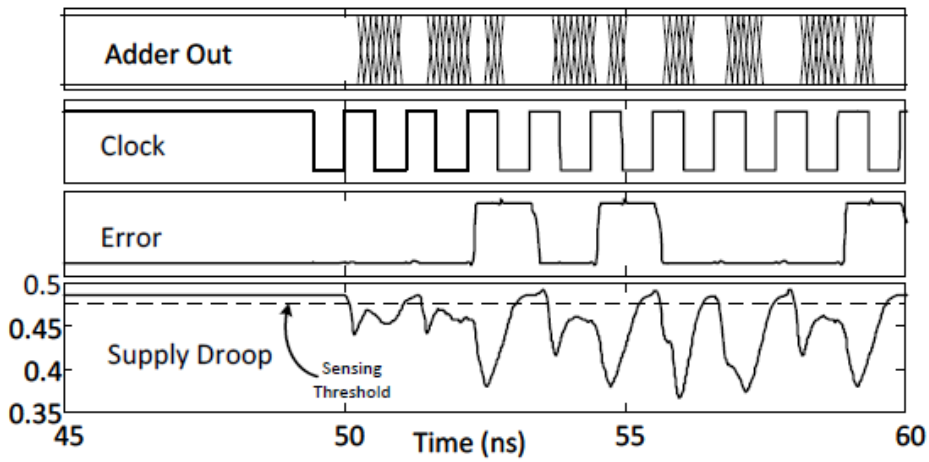
Figure 2.2. Original concept of a CSCD device (ref. [10]).

The current variations are inputted into I_{in} and the BJT plays a role of current amplifier, allowing the current variation to be translated in V_{out}. The output V_{out} is then compared to a threshold value: if V_{out} is superior to that threshold value, a digital bit is set to 1 to show that the computation is not complete yet. Other techniques allow to use a CMOS device instead of the BJT and use more or less advanced tricks to compare the current to a threshold (see [10] for an example using inverters).

More recently, CSCD devices have been studied to be used in synchronous circuits at near threshold voltage in [11]. Figure 2.3 is the design of such a device. It produces a digital signal error when the current is above some threshold at the rising edge of the clock.



(a)



(b)

Figure 2.3 (a) CSCD circuit in a synchronous circuit.

(b) CSCD timing diagram (ref. [11]).

The noise filter serves as limiting the supply voltage variations. Between the logic and the supply voltage a gate transistor plays the role previously described with Figure 2.2. The drain and the source signals of that transistor are then sent to the two ports in and $\overline{in}$. Then, by adding a certain amount of current to the node Di thanks to the transistors mid*i*, we set the threshold value to which we want to compare our current. Then at each rising edge of the clock, the output of a latch is set or reset according to the comparison between Di and its negation.

In our work we considered the following simplified circuit (figure 2.4). In the offset-calibrated quantizer block, we replaced the transistors setting the threshold by a current source.

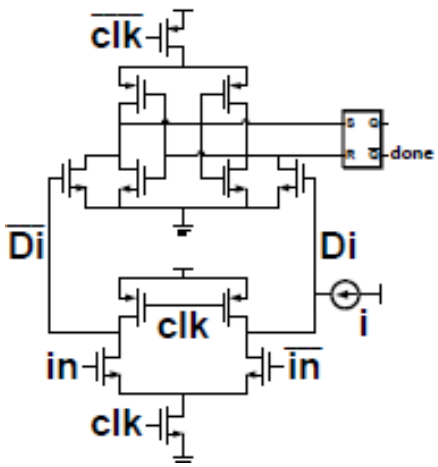


Figure 2.4. Simplified version of the offset-calibrated quantizer block.

We see that the current *I* added to the branch Di allows us to set the threshold value of the CSCD device. In the original design, the value of that current could be set digitally by activating or deactivating the mid*i* transistors.

Finally, note that since we didn't consider supply voltage variation in our study, the noise filter is not necessary.

Chapter 3

Proposed approach

We describe here more precisely our technique to automatically select the supply voltage according to process variation. A schematic description of our approach is described in Figure 3.1.

We use our CSCD circuit as designed by [11] by checking at rising edge of the clock if our computation is complete or not; if it is incomplete, then the error signal raises. When we detect the error signal to 1, we increase the supply voltage of the overall circuit, until reaching a state where no error is detected.

This suppose that the circuit is well designed enough to have *at least* one voltage which allows the circuit to be error free. In our approach, if this not the case, then errors will keep raising as it was impossible to have a circuit without any error.

Also, particular attention need to be done to the threshold value that we use for the CSCD sensor. Such a value depends of the supply voltage, and need to be

updated when we increase the supply voltage: the supply voltage managing circuit need to give a feedback to the CSCD circuit (V_i in figure 4.1).

This also means that we need to find a threshold value at a given voltage which ensures that the CSCD output is correct for *all* variations that we want to cover. In the traditional approach, the designer will define a set of intrinsic and extrinsic variations where he wants his circuit to behave correctly and check that the timing constraint is valid for those variations. In our approach, he will have to set the threshold value of the CSCD device instead. However, once the threshold values set, the circuit will be able to save power by adapting itself to the defined set of variations.

A final remark concerning the CSCD is that, for most designs, it should be made using low threshold cells (except of course for the latch which is like a digital circuit). If we lower down the supply voltage too close from the threshold voltage of the analog part of the CSCD device, the device will not behave has intended. Using low voltage thresholds cells allows to lower down the supply voltage for the logic block much, allowing more energy savings at the end.

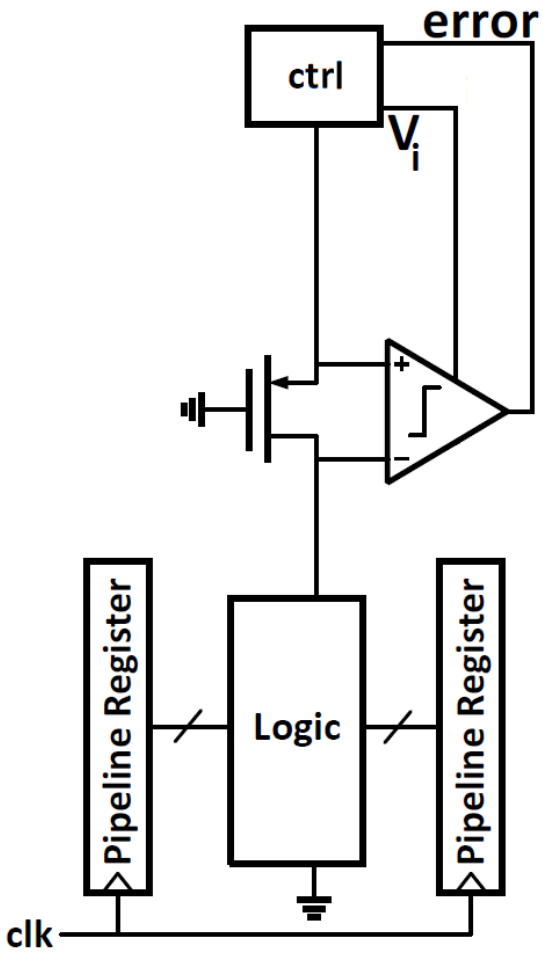


Figure 3.1. Schematic description of our approach.

In the descriptive schematic at figure 4.1, the CSCD is connected at the drain and the source of a power gate used in our logic block. The error signal is transmitted to a control block (ctrl), which selects the right supply voltage to send to the logic

block and sets the threshold value of the CSCD device (signal V_i).

There are several ways to design the ctrl block. If the range of possible supply voltage is small, we can supply several power rails to the block and then choose which one to apply. In that case, the power consumption and area of the block is small (compared to large logic blocks) since it only contains some digital gates, but we might have to handle a relatively import quantity of power rails if we want to optimize too many logic blocks.

Another option is to embed a voltage level shifter in the logic block and feed it with the nominal supply voltage. Recent advances in voltage level shifter (see for example [14]) indicated that such an option would be available in a near future. Furthermore, our technique coupling the CSCD and the control block could be associated with multiple logic blocks at the same time: if two logic blocks are physically close on the chip (which is usually the case), the difference of variations between those two will be very low and our technique would increase the supply voltage when variations make the transistors slower on the 2 blocks and increase it in the opposite case. This would of course work for *n* blocks close enough.

Chapter 4

Experimental setup

We simulated a full 32-bit adder in HSPICE using a transistor model from Synopsys University program interoperability PDK 28/32nm. According to the Synopsys documentation, the model has “characteristics similar to […] real foundries (TSMC28 or IBM32)” ([13]).

4.1 Intrinsic variations

Synopsys design kit provides process corners to allow process variations. We tested our design in the FF, TT, SS corners in order to have respectively the best, average and worst power saving cases.

As the model doesn’t define the standard deviation of parameters influencing on the process variation (oxide thickness, gate transistor width, threshold voltage), we couldn’t run a Monte-Carlo analysis on our circuit.

However, according to Synopsys documentation [13], the process corners define the extreme cases in process variation (“by changing the threshold voltage

(vth0) and oxide thickness (tox) in the range of +/-5%", [13]), so we can approximate without exaggeration that benchmarking our technique on the FF and SS corners allows to have the best and worst case power savings.

4.2 Extrinsic variations

In our work, we only considered temperature variation. Although other variations such as aging could be considered, we chose to consider temperature variations from -20°C to 50°C.

It should be noted that if other extrinsic variations are considered, then the power savings of our techniques increase or is the same: more variations means increased supply voltage in the traditional approach and on the other hand gives us more flexibility.

4.3 Control block

The voltage correction block (control block) could be designed as a simple, small, energy efficient digital electronic circuit. However, for the sake of simplicity, we chose to script the behavior of such a circuit.

4.4 Logic block for testing

We designed a 32-bit adder on HSPICE for the purpose of testing the circuit. Note that a designer could first produce a RTL design of his digital circuit, then export

it to HSPICE through tools such as Synopsys Design Compiler. A mixed signal-simulation using Synopsys Custom Designer would also be possible, as our technique can easily be transposed to bigger designs.

The schematic of our adder is given in Figure 5.1. We associated the different full-adder as a ripple carry adder.

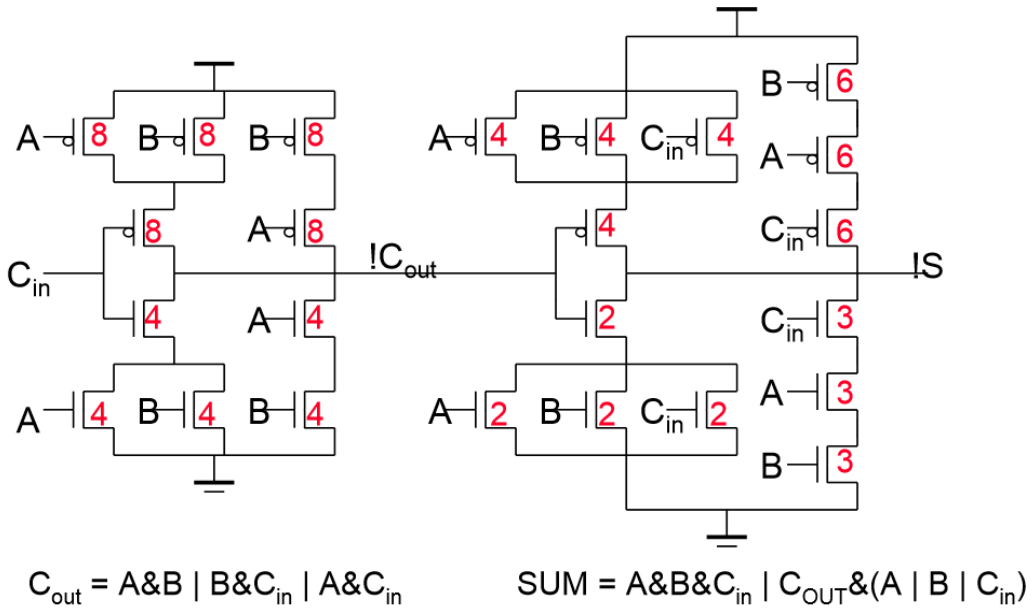


Figure 4.1. Schematic of the 1-bit full adder used (in red transistor sizing).

One full-adder consists of 24 transistors, plus 4 transistors for inverting the outputs.

4.5 Experimental parameters

Our experiment parameters are summarized in table 5.1.

Table 4.1 Experiment parameters

Parameter name	Data
Technology node used	28/32nm
Transistors nominal supply voltage	1.05V
Variation of Logic block supply voltage	0.5V ~ 1V
Step size increase in supply voltage	0.05V
Frequency	200MhZ
Supply voltage satisfying frequency @TT, 25°C	0.75V

Chapter 5

Experiment results

We first describe the response of our circuit by only considering intrinsic variation. In a second time. We then consider temperature variation to finally deduce the gain in power for each case.

Note that we fed the adder with random input, inserting at multiple points in the input data to cause a critical path in the adder, thus causing an increase in supply voltage. In real conditions, such an input wouldn’t occur in such a short time lapse (thus causing even better power performance). Subsections 5.1 to 5.3 simulations are realized at the temperature of 25°C.

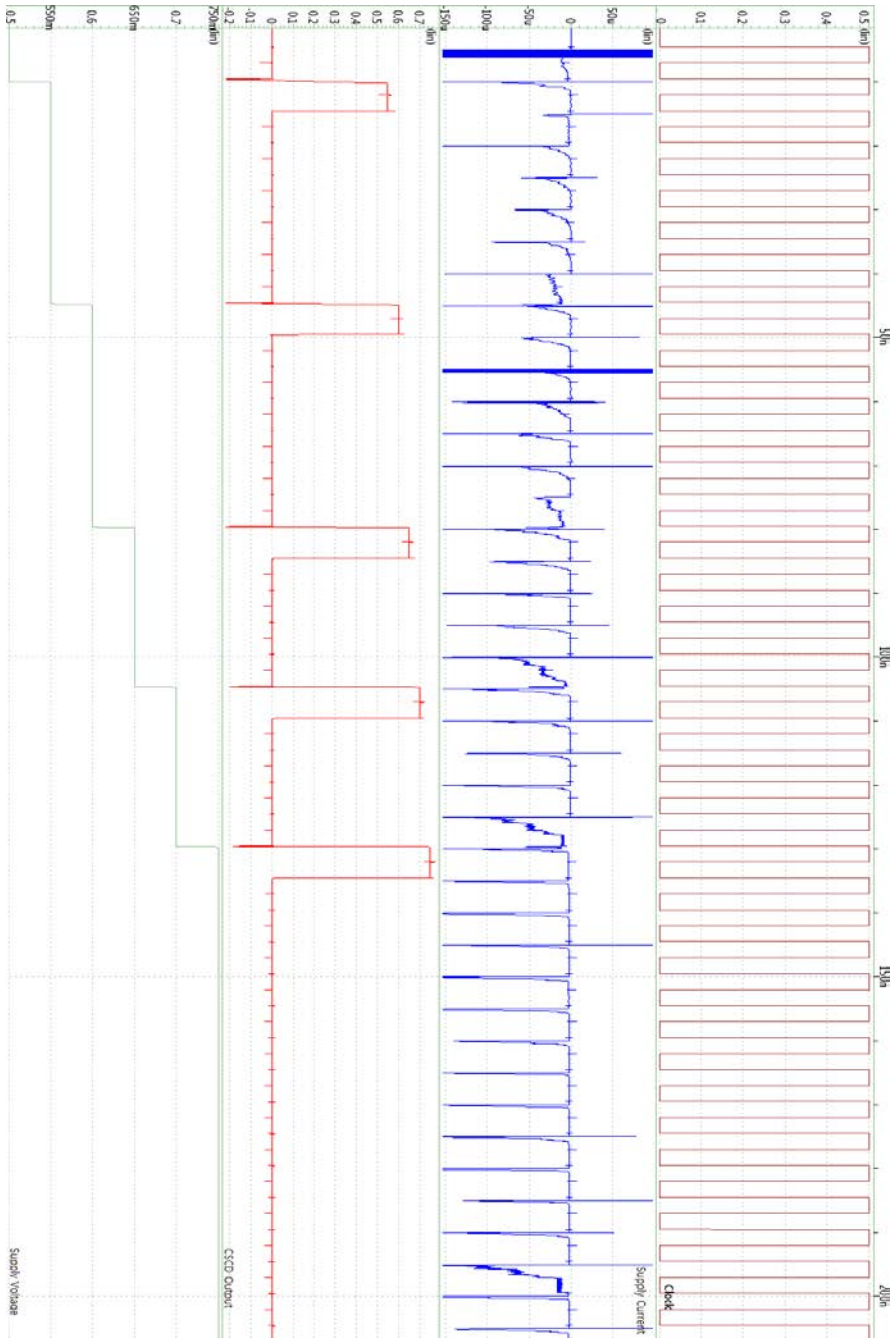


Figure 5.1 Timing diagram of our circuit at TT. First line is the 200MHZ Clock, second the current consumption, third the CSCD output, fourth supply voltage.

5.1 Results at the TT corner

We chose the clock frequency so that at the TT corner (no process variation) the supply voltage is 0.75V. Nevertheless, if the designer make such a choice, our technique will start from 0.5V supply voltage, and then be increased to 0.75V, as we can see in Figure 5.1.

5.2 Results at the FF corner

At the FF corner, the adder circuit is able to go at a faster speed, thus decreasing the required supply voltage to operate at 200MHz. The FF corner is for our technique the optimum case. We see on the Figure 5.2 that the FF corner allow to reduce the supply voltage of our adder to 0.6V.

5.3 Results at the SS corner

At the SS corner, the adder circuit has to go at a slower speed, thus increasing the required supply voltage to operate at 200MHz. This is for us the worst case.

In a classical design, the designer would have to set the supply voltage of the chip at the result of this simulation: 0.95V (figure 5.3). It means that by using our technique, in the SS case the supply voltage cannot be reduced; but it also means that compared to the traditional approach, we can lower down the voltage to 0.75V at the TT corner and 0.6V at FF corner, which reduced respectively the supply voltage of **0.20V** and **0.35V**.

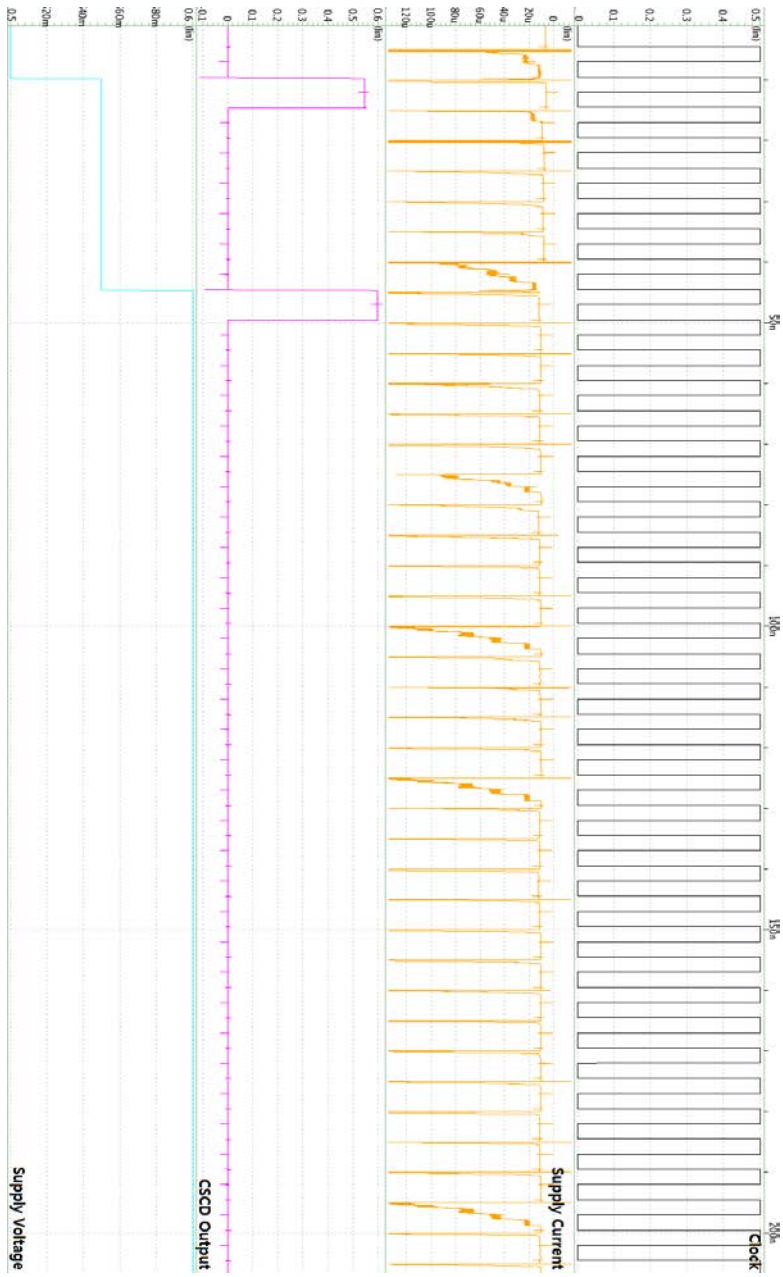


Figure 5.2 Timing diagram of our circuit at FF. First line is the 200MHZ Clock, second the current consumption, third the CSCD output, fourth supply voltage.

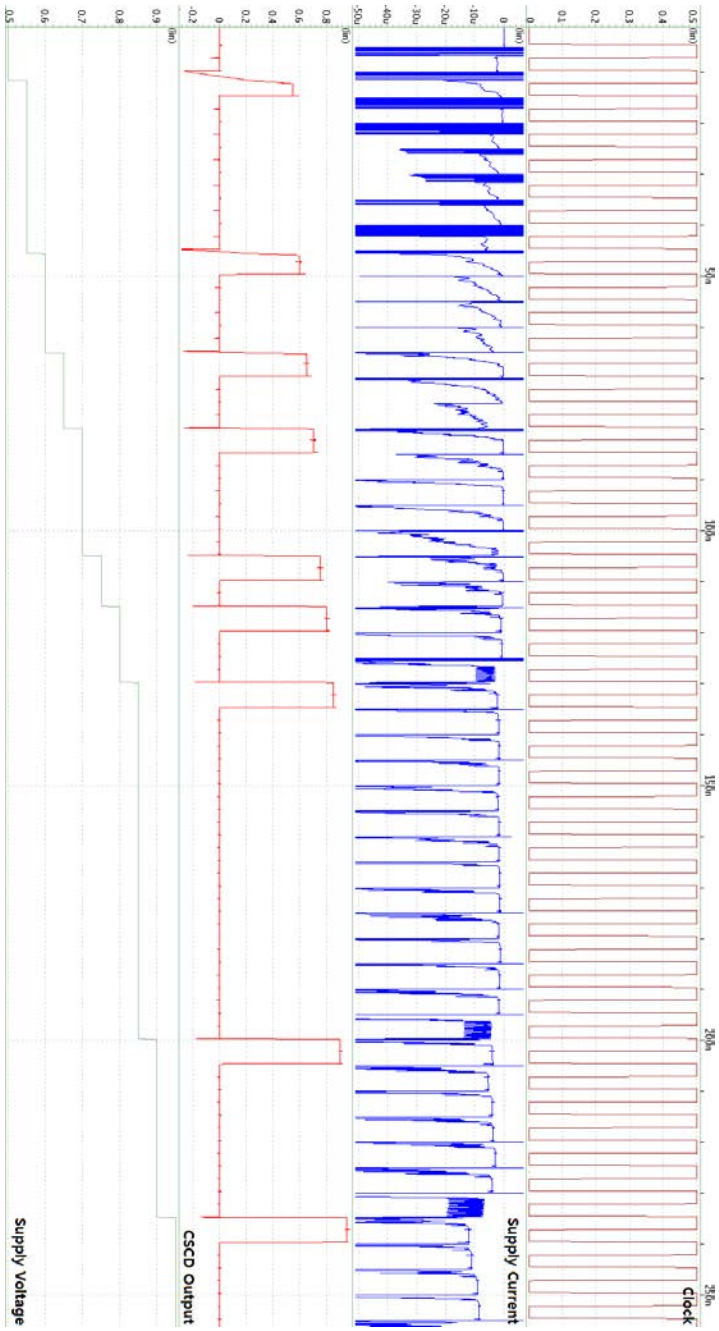


Figure 5.3 Timing diagram of our circuit at SS. First line is the 200MHZ Clock, second the current consumption, third the CSCD output, fourth supply voltage.

5.4 Effects of temperature variation

During our previous simulations, we didn't take into account the effect of temperature variation. In every corner tested, we have to take into account the temperature variation to fully simulate most of the variations possible in a chip.

For every corner, we simulated temperature variation from -20°C to +50°C. We plotted results of the simulations in figure 5.4. We see that for the FF corner, the temperature variations allows the chip to have a supply voltage of 0.55V under 18°C, thus allowing even more power saving.

More importantly, we remark that for the SS corner at higher temperature, the supply voltage has to be 1V for the chip to meet its frequency. It means than in a traditional design, the designer should use a reference supply voltage of 1V to meet its frequency target (instead of the previously clamed 0.95V value which wasn't considering temperature variation).

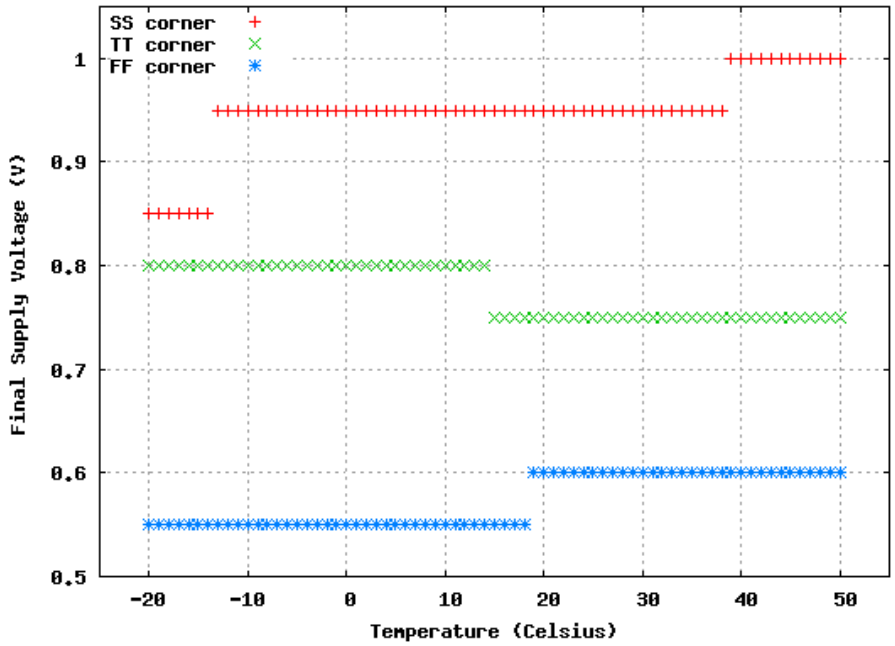


Figure 5.4 Supply voltage evolution of different corners for temperature variations.

5.5 Final power savings

With our technique, by only considering process and temperature variations, we were able to reduce the supply voltage from 1V in the traditional approach to 0.75V in TT corner and to 0.6V in FF corner. In FF corner at low temperature, we were even able to reach 0.55V final supply voltage: we roughly divided the supply voltage by two. We can summarize our voltages reduction in the following table.

Table 5.1 Supply voltage reductions

Corner	Supply voltage savings
SS (with T variations)	0V~0.05V
SS (without T variations)	0.05V
TT (with T variations)	0.20V~0.25V
TT (without T variations)	0.25V
FF (with T variations)	0.40V~0.45V
FF (without T variations)	0.40V

However, this does not directly translate into power savings. Using the CSCD has a cost, especially since we used low leakage cells. In the following figure 6.5, we compared how much power we saved by taking into account the cost of using the CSCD. Note that we neglected the power cost of the additional logic used to set the voltage, as this can be considered small compared to huge logic blocks.

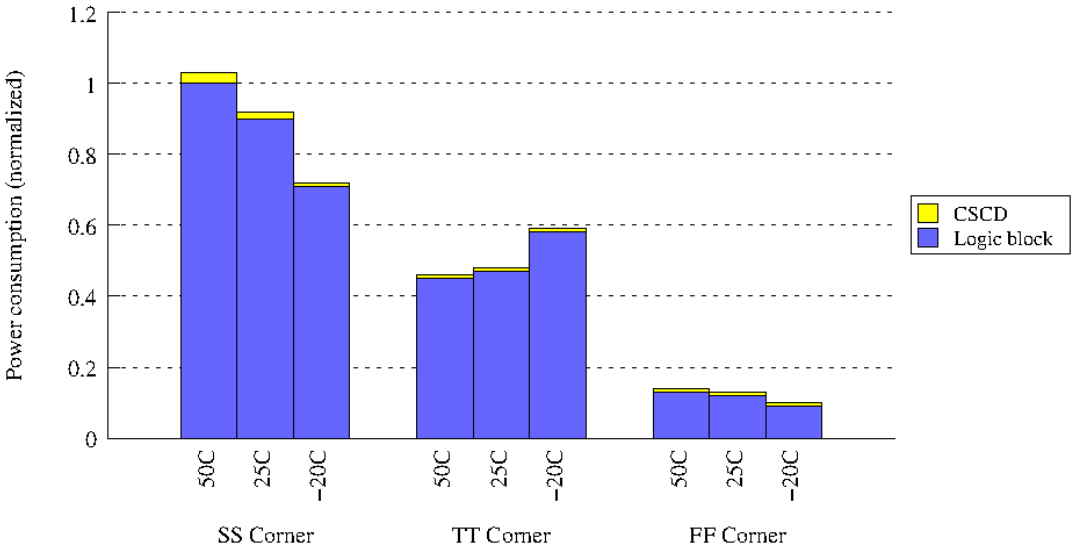


Figure 5.5 Power consumption of our circuit, normalized to the classic approach using the same variation parameter for each comparison.

For each possible variation, we computed the power consumption in the classic approach (over-increasing the supply voltage to meet the timing constraint) and our technique. We see that at worst, our technique increases power consumption by 3% (consumption of the CSCD circuit), at best it is reduced by 90%, and on the average case (TT corner) at 25°C we saved 52% power.

Also, since our logic block is fairly small, and the increase of power consumption due to CSCD is most of the time around 1% (in every case less than 1%), we've shown that with a circuit large enough (bigger than a 32-bit adder), the power consumption of the CSCD is negligible.

Chapter 6

Conclusion and future work

We’ve devised a technique which allows to dynamically select the minimum supply voltage at runtime to match a timing constraint. For that purpose, we used a Current-Sensing Completion Device to detect when a computation is unfinished at the beginning of a clock cycle, and raised the voltage step-by-step s in that case.

We saw that by only considering temperature and process variation, our technique saves 52% of power on average and 90% in the best case. In the worst one, the power consumption is only increased by 3%. Furthermore, by considering other variations (such as aging) our technique could perform even better. This could be an extension of our work.

Also, in our model, we didn’t consider the increased in area and/or power to supply the logic block with multi-Vdd domain. However, how we stressed earlier, a voltage level shifter could be used with our control block. Another extension of our work could be to include level shifters in our model and include their power consumption in our final result.

Bibliography

[1] Dreslinski, Ronald G., et al. "Near-threshold computing: Reclaiming moore's law through energy efficient integrated circuits." *Proceedings of the IEEE* 98.2 (2010): 253-266.

[2] R. Swanson and J. Meindl, B Ion-implanted complementary MOS transistors in low-voltage circuits, [IEEE J. Solid-StateCircuits, vol. 7, no. 2, pp. 146–153, 1972 Borkar, Shekhar, et al.

[3] "Parameter variations and impact on circuits and microarchitecture." *Proceedings of the 40th annual Design Automation Conference*. ACM, 2003.

[4] Calhoun, Benton H., Alice Wang, and Anantha Chandrakasan. "Modeling and sizing for minimum energy operation in subthreshold circuits." *Solid-State Circuits, IEEE Journal of* 40.9 (2005): 1778-1786.

[5] Fick, David, et al. "Centip3De: A cluster-based NTC architecture with 64 ARM Cortex-M3 cores in 3D stacked 130 nm CMOS." *Solid-State Circuits, IEEE Journal of* 48.1 (2013): 104-117.

[6] Dreslinkski, Ronald G., et al. "An energy efficient parallel architecture using near threshold operation." *Proceedings of the 16th International Conference on Parallel Architecture and Compilation Techniques*. IEEE Computer Society, 2007.

[7] Agwa, Shady, Eslam Yahya, and Yehea Ismail. "Design techniques for variability mitigation." International Journal of Circuits and Architecture Design 1.1 (2013): 20-40.

[8] Dean, Mark E., David L. Dill, and Mark Horowitz. "Self-timed logic using current-sensing completion detection (CSCD)." *Asynchronous circuit design for VLSI signal processing*. Springer US, 1994. 7-16.

[9] Akgun, Omer Can, Joachim Neves Rodrigues, and Jens Sparsø. "Energy-minimum sub-threshold self-timed circuits using current-sensing completion detection." *Computers & Digital Techniques, IET* 5.4 (2011): 342-353.

[10]Nagy, Lukas, and Viera Stopjakova. "Current Sensing Completion Detection in deep sub-micron technologies." Design and Diagnostics of Electronic Circuits and Systems (DDECS), 2010 IEEE 13th International Symposium on. IEEE, 2010.

[11]Crop, Joseph, Robert Pawlowski, and Patrick Chiang. "Regaining throughput using completion detection for error-resilient, near-threshold logic." Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE. IEEE, 2012.

[12]Mitigating the Effects of Process Variation in Ultra-low Voltage Chip Multiprocessors using Dual Supply Voltages and Half-Speed Units

[13]SAED 32/28nm Spice Model Documentation, Spice Manual

[14]Burton, Edward A., et al. "FIVR—Fully integrated voltage regulators on 4th generation Intel® Core™ SoCs." Applied Power Electronics Conference and Exposition (APEC), 2014 Twenty-Ninth Annual IEEE. IEEE, 2014

국문 초록

현대의 내장형 시스템은 점점 더 전력 소비에 대한 제약을 많이 받고 있다. 더욱 많은 데이터를 더욱 빠른 속도로 처리해야 하는 요구가 있음에도 불구하고 기기의 신뢰성과 배터리 수명을 유지하기 위해서 에너지 소비를 낮추는 것이 필수적이다.

전력 개폐, 클럭 개폐, 등과 같이 칩의 전력 소비를 낮추는 많은 기술이 있지만 그 중에서도 공급전압을 낮추는 것(칩의 주파수도 함께 낮추게 될 수 있음)이 가장 효과적인 것으로 알려져 있다. 그러나 공급전압을 트랜지스터의 문턱전압에 가까울 정도로 많이 낮추게 되면 내부 또는 외부의 변이(공정 변이, 온도, 노화, 등등)에 따라 논리회로의 지연시간이 지수적으로 변하게 되고, 따라서 설계자가 더욱 증가된 시간 여유분을 두게 만든다.

본 논문은 논리회로의 속도가 주어진 시간제약에 맞도록 공급전압을 자동적으로 조절해 주는 기술을 제안한다. 이는 공정이나 온도 변이에 따라 설계자가 설정한 시간 제약을 만족하는 범위에서 최소의 공급전압을 선택하는 기술이다. 이 기술은 기본적으로 가해지는 논리회로의 공급전압을 줄여서 소비전력을 줄일 수 있게 해 준다. 28/32nm 기술 노드에서 실험한 결과,

제안하는 방법으로 공급전압을 0.55V에서 1V 사이에서 제어함으로써 공칭 공급전압이 1.5V인 논리회로의 소비전력을 평균 52% 줄이는 데 성공했다.

주요어 : 상위수준합성, 5 개 이내

학번 : 2014-22514