High Performance Digital Circuit Techniques

by

Sayed Alireza Sadrossadat

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2009

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Achieving high performance is one of the most difficult challenges in designing digital circuits. Flip-flops and adders are key blocks in most digital systems and must therefore be designed to yield highest performance. In this thesis, a new high performance serial adder is developed while power consumption is attained. Also, a statistical framework for the design of flip-flops is introduced that ensures that such sequential circuits meet timing yield under performance criteria.

Firstly, a high performance serial adder is developed. The new adder is based on the idea of having a constant delay for the addition of two operands. While conventional adders exhibit logarithmic delay, the proposed adder works at a constant delay order. In addition, the new adder's hardware complexity is in a linear order with the word length, which consequently exhibits less area and power consumption as compared to conventional high performance adders. The thesis demonstrates the underlying algorithm used for the new adder and followed by simulation results.

Secondly, this thesis presents a statistical framework for the design of flip-flops under process variations in order to maximize their timing yield. In nanometer CMOS technologies, process variations significantly impact the timing performance of sequential circuits which may eventually cause their malfunction. Therefore, developing a framework for designing such circuits is inevitable. Our framework generates the values of the nominal design parameters; i.e., the size of gates and transmission gates of flip-flop such that maximum timing yield is achieved for flip-flops. While previous works focused on improving the yield of

flip-flops, less research was done to improve the timing yield in the presence of process variations.

Acknowledgments

First of all, I would like to thank my supervisor, Professor Mohab Anis and also Professor Kumaraswamy Ponnambalam for their constant support, encouragement and guidance throughout my Master's program. It was my honor to work under their supervision and guidance. I am thankful to Professor Andrew Morton and Professor M. Anwar Hasan, the readers of my thesis, for their invaluable suggestions and corrections.

I am also grateful to my friends and my colleagues for their encouragement when I needed those the most. Also, I would like to have the best thanksgiving to my parents for their faith in me during my stay abroad.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

1.1.1 Importance of high performance computing

From a system designer's perspective, the performance of a digital circuit expresses its computational ability. For instance, a microprocessor often is characterized by the number of instructions it can execute per second. The performance metric depends both on the architecture of a processor - for instance, the number of instructions it can execute in parallel - and the actual design of logic circuitry. When focusing on the pure design, performance is most often expressed by the duration of the clock period (*clock cycle time*), or its rate (*clock frequency*). There are lots of efforts that were done to gain high performance in digital circuits. Different methods for achieving high performance consist of algorithm level methods and circuit level methods. There are different kinds of algorithm and circuit level methods that typically target combinational and sequential circuits. As combinational circuits, adders which are used widely in digital systems are investigated in this thesis to achieve high performance. As sequential circuits, the design of high performance flip-flops is probed.

1.1.2 High performance serial adders

In microprocessors, the addition operation is used more than any other operations such as multiplication and division. Also, serial implementation is an important class of circuit design that consists of numerous applications including signal processing. Serial architectures, which have serial data transmission and serial computation, are pivotal in the design and implementation of VLSI signal processors and the implementation of special-purpose processors [1] and therefore are pivotal for getting high performance in several processors. In signal processing, where high performance is crucial, the architectures are particularly appropriate [1] due to serial data transmission, performance constraints, and implementation costs. Generally, serial techniques are a composition of serial data transmissions and serial calculations. When operands are serially received, during calculations, the intermediate results pass to the arithmetic units serially, and mostly digit by digit [2]. In [3], the analysis, provided by the simulation results, shows that serial adders dissipate less power than the other ones and therefore this architecture can be used to improve the performance of processors. Also, it is shown that serial adders achieve a better Power-Delay-Product (PDP) and Energy-Delay-Product (EDP) than other adders in nanometer technology [3]. It is reported in [4] that serial adders might be better for low-power operations with redundancy for enhancing reliability and for the goal of performance achievement, and therefore, this architecture was chosen to be probed.

The primary problem of serial techniques is that they have higher delay than that of other techniques. However serial techniques operate in lower power dissipation. This problem motivates the development of a method for solving the delay problem of serial additions to get higher performance in digital circuits. The new method is faster than the fastest known serial addition technique, the Kogge Stone adder, and consumes less power and is easy to implement.

1.1.3 High performance flip-flops under process variations

In microprocessor and digital circuits, flip-flops are indisputably important circuit blocks that are investigated here in order to achieve higher performance consequently resulting in the higher performance of microprocessors and digital circuits. On the other hand, variability in the device process parameters are becoming more and more significant in nanometer technology. The growth of variability can be attributed to multiple factors, including the control of the manufacturing specifications, the emergence of new systematic variation-generating mechanisms, and most importantly, the increase in fundamental atomic-scale randomness such as variations in the number of dopants in transistor channels [20]. Process variations impact the performance of devices and consequently circuits. Therefore, timing yield loss due to variability has become a challenging issue in nanometer technology. Yield loss augments design costs because dies that do not meet performance requirements are rejected [21]. Circuits such as flip-flops that are designed small are highly sensitive to process variations [22]. Moreover, flip-flops should be designed keeping in mind that they embody a large percentage of a chip's area. Because of the increased transistor density in modern technology nodes (assuming flip-flops take up approximately 10 % of the chip's area), large chips such as microprocessors or general purpose units (GPUs) must hold at least a million flip-flops. Similar to statistical SRAM design, uncertainties in the flip-flop timing parameters require the assignment of appropriate safety margins. Therefore, the timing variability overhead of flip-flops directly impacts the fastest achievable operational frequency in the design. Thus, finding ways to limit the variability of flip-flop timing parameters is beneficial in terms of performance and area constraints.

1.2 List of Contributions

1.2.1 Design of a high performance constant delay serial adder

In this thesis, we propose a new high performance serial adder that achieves higher performance than conventional methods. Therefore, the problem of the latency in serial additions can be improved. The adder is based on the idea of a Constant-Delay (CD) during the addition of operands, and will be shown to be superior over the Kogge-Stone (KS). Also this method can be applied for multiplication and parallel structures. We show that the new adder exhibits low hardware complexity and power consumption.

1.2.2 Develop a framework for designing flip-flops under process variations

In this thesis, a statistical framework for designing flip-flops under process variations is introduced. The framework reports values of transistors widths that maximize timing yield under area and performance constraints.

Chapter 2

Background

2.1 High performance adders

2.1.1 Algorithms and different kinds of adders

Adders are pivotal arithmetic blocks in most of digital circuits and they are built with different algorithms that have different performances. All of the different structures have a delay dependency on the bit-width of the operands, i.e., the longer the operands, the later the results of the addition are ready, and the more area and power. Therefore there is a tradeoff between performance and area. Some structures perform the addition in a time linearly proportional to the bit-width of the operands. The result is a small hardware-area production. Other structure's delay that can be achieved for the addition of two operands is logarithmically dependant on the bit-width of the operands. The delay reduction results in hardware complexity, and conveys that other aspects, area and power, should be studied.

In [6], one of the earliest algorithms is introduced where a tree structure (logarithmic delay) is used to do addition. In [7], two digit serial adders are compared, one processes the operand digits and produces the least significant digit first, and the other processes the operand digits and produces the most significant digit first. In [8], a fast addition algorithm which they called ELM, a tree of simple processors, requiring $O(log_2(n))$ execution time is used. Here, n is the word length. In [9], the shuffled serial adder is introduced and is derived from the Kogge-Stone adder. The latency of this adder is proportional to $log_2(n)$, where is the length of the operands. Furthermore, a generalized earliest-first (GEF) addition algorithm is reported in [10]. This algorithm accelerates the carry propagation addition (CPA) and schedules bit-level operations of the CPA in an earliest-first manner to reduce the overall delay. In [11] a simple network for the serial addition or subtraction of BCD numbers is described. The architecture contains a binary adder and a correction system with another adder.

In the look-ahead type adders, the carries of each column are calculated by using the values of the operands and only the input carry. In this type of adder, the carry of each column is independent of the carries of the previous columns, and is calculated separately. There are also logarithmic look-ahead adders that use the look-ahead concept in a logarithmic manner, and produce the carry look-ahead trees for conducting the addition [5].

In [16], a class of zero-deficiency prefix circuits which provide the minimum depth for a given width, is discussed. Some researchers have saved one-logic level of implementation, resulting in a faster and also higher performance for the parallel-prefix addition [17].

Sparse trees are introduced to deliver the reduction in the hardware and area and therefore, some degradation in performance. In [5], [14], the Brent-Kung adder is a well-known structure of the look-ahead group that is optimized in terms of the area [15]. In [5], the hybrid Kogge-Stone/ Brent-Kung adders are examined regarding the tradeoffs between the hardware-area and the delay of operations and performance. The Kogge-Stone adder is a member of the look-ahead group. The Kogge-Stone tree requires much hardware area and many transistors [5], [12], [13] and [15].

In [18], a performance evaluation analysis was carried out between flagged prefix adders with other well-known prefix adders. Here, the Kogge-Stone adder is the fastest with the highest performance and the Brent-Kung adder is the most efficient in terms of power dissipation. Also, high performance VLSI adders have been compared in the Energy-Delay space so that the appropriate selection is possible at the beginning of the design process [19].

2.1.2 Kogge-Stone adder

The Kogge-Stone (KS) adder, one of the fastest adders, is now described in detail to show how it can make addition faster. However, it sacrifices the area proficiency goal. The Kogge-Stone adder comprises blocks: Generate (G), Propagate (P), and Dot (D) operations. Generate block shows whether the input bit columns which are given to this block, can generate the carry for the next column or not. Also, Propagate block determines if the input bits that are related to the special columns of bits, given to this block, can propagate a carry to the next column of bit or not. D block is a composition of the P and G blocks of two subsequent stages and produces the P and G products for the other stages which are shown in Fig. 2.1 and according these definitions,

$$P_{Stage3} = P_{Stage1} \cdot P_{Stage2} \tag{2.1}$$

and

$$G_{Stage3} = G_{Stage1} \cdot P_{Stage2} + G_{Stage2}. \tag{2.2}$$

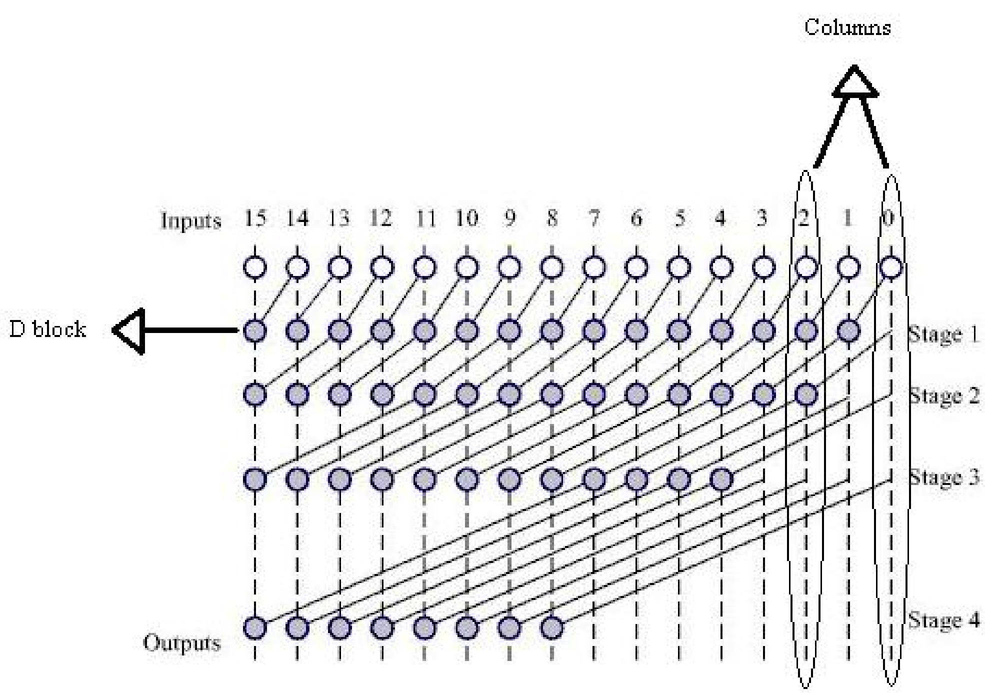


Figure 2.1: Kogge-Stone adder

For adding two n-bit numbers (A and B) by using Kogge-Stone (KS), first stage produces $G_0, G_1, ..., G_{n-1}$ and $P_0, P_1, ..., P_{n-1}$ and intermediate sums, that are $S_0, ..., S_{n-1}$ ($S_i = A_{ii}$), by using the bits of the two numbers. Then, the other $\log_2(n)$ stages are used that each stage contains many D operations and combine the different P and G of the previous stages. In the last stage, the final G and P products are generated to find the result of addition of two numbers such that:

Final G products: $G_{[1-0]}, G_{[2-0]}, ..., G_{[(n-1)-0]}$

and

Final P products: $P_{[1-0]}, P_{[2-0]}, ..., P_{[(n-1)-0]}$

Now, the final carry input, arriving at each column, is calculated by combining these G and P products with C_{in} by using G block, where C_{in}, is connected to its input. Then, for the last stage, the final sum of each column is computed by using XOR gate. The inputs of these gates are the final carries of the previous stage and the intermediate sums. Fig. 2.1 is a graph of a Kogge-Stone adder that has the minimum logic depth, resulting in a fast adder but with a large area. In this figure, the intermediate nodes represent the D blocks. For doing serial addition, several numbers of this adder should be used serially using a clock signal that in each clock, a new number will be added by the result of the additions of the previous clocks.

The proposed CD method is compared with Kogge-Stone method in this thesis because the Kogge-Stone (KS) method yields a delay in the order of a logarithmic function of the length of the words and because it is optimized in the timing issues and is state of the art for high performance adders [3], [15]. However it requires much area and power consumption.

The proposed method in this thesis is not only independent of the bit-width of the operands (except the final converter) but significantly lower in delay, compared to the fastest known ones so it can result in a better performance for digital circuits.

2.2 High performance flip-flops under process variations

2.2.1 Process variations in the flip-flops and their impact on the performance

Flip-flops play a pivotal role in the arithmetic blocks in most of digital circuits. By the scaling of the flip-flops in nanometer technology, process variation becomes more important in the performance of the flip-flop.

There are several sources of variations in the transistors of the flip-flops such as random dopant fluctuation (RDF), channel length variation, line edge roughness (LER), gate oxide thickness variations, oxide charge variations and channel width variations. These parameters affect the threshold voltages of the transistors and consequently can have impact on the current of the transistors. Variations in the current of transistors can significantly impact on the performance and power consumption of the flip-flops.

Because of the important role of flip-flops in timing behavior of sequential designs and their performance, there has been a lot of research to increase design robustness of flip-flops against variations. Therefore, the thrust has been to modify a flip-flop's architecture in order to ensure robustness against noise and soft errors. However, less effort has been made to mitigate the effects of process variations on flip-flops.

2.2.2 Statistical methods for improving flip-flop performance

A statistical analysis method has been proposed in [23] to extract the probability mass function of the flip-flop's setup-time and hold-time. Also, clock skew scheduling is a technique that improves the operational frequency [24]. Typically, statistical skew schedulers determine the relative clock arrival time to each register to enhance the clock period [25] and timing yield loss [26] due to timing constraints violations. However, skew scheduled designs are more sensitive to unpredictable variations because of tight slacks in the combinational paths. Moreover, clock tree construction is more difficult for clock skew scheduled designs than zero skewed designs. Another approach to address delay variations in the circuit is to use latch-based designs. Latches are more tolerant to delay variations, because they have no hard boundary and are transparent for half a clock period. In [27], a clock scheduling for latches has been presented to improve the timing yield. However, latch-based designs do not satisfy digital designers, because such designs usually need two separate clocks which mean a significant power and area overhead. Furthermore, generating two non-overlapping clocks can be difficult in high performance design.

2.2.3 Different types of flip-flops

There are several kinds of flip-flops. An edge-triggered flip-flop samples the data input on one edge of the clock and keeps the sampled data on the output during the remainder of the clock period. A simple Master-Slave flip-flop can be constructed from two cascaded level-sensitive latches, as shown in Fig. 2.2. When the clock signal is low the first latch, called master latch, is transparent and the input is transferred to the intermediate node (X). The second latch, called the slave latch, is opaque so the output (Q) is held at its previous state. When the clock signal makes a low-to-high transition, the master latch becomes opaque, and the slave latch becomes transparent, and the intermediate data at (X) is transferred to the output (Q). The data on the output is valid for the remainder of the clock period [28].

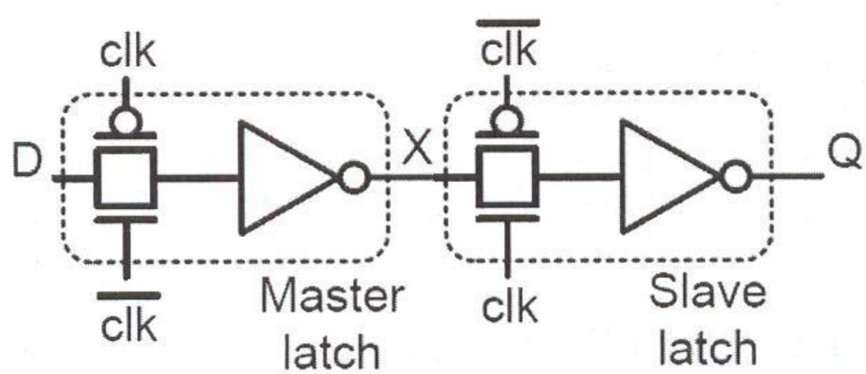


Figure 2.2: Schematic example of a positive edge-triggered flip-flop [38]

.

A timing diagram of a positive edge-triggered flip-flop is shown in Fig. 2.3 in which, *Setup-Time (T_{setup})* is defined as the minimum time that the input data should be available before the clock sampling edge arrival, *Hold-Time (T_{hold})* is defined as the minimum time that the input data should be available after the clock sampling edge arrival, *Clock-to-output delay (T_{Clk-Q})* represents the delay from the sampling clock edge (Clk) to the time, the latched data is valid at the output (Q), *Data-to-output delay (T_{D-Q})* represents the delay from a transition of the input data (D) to the time, the latched data is valid at the output (Q) [29].

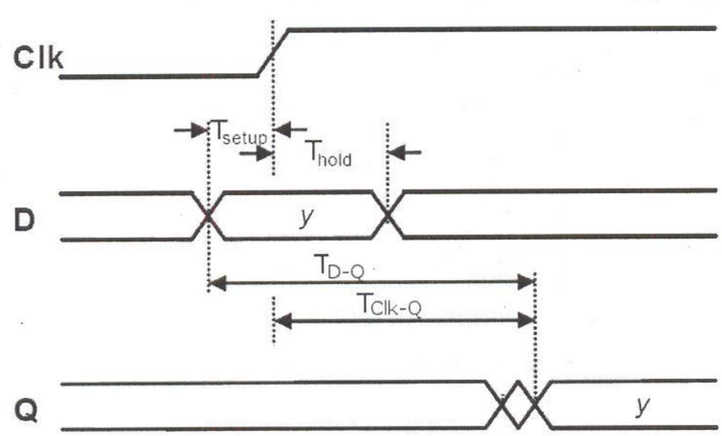


Figure 2.3: Timing characteristics for a positive edge-triggered flip-flop [38]

.

Another kind of flip-flop is sense-amplifier based flip-flop, which is shown in Fig. 2.4, utilizes a sense-amplifier to sample the data [38, 28]. The advantage with the sense-amplifier flip-flop is the low number of clock transistors, which gives low clock load. One of the important drawbacks with the sense-amplifier flip-flop is the pre-charged behavior of the sample stage, which is power-consuming especially when the data activity on the inputs is low.

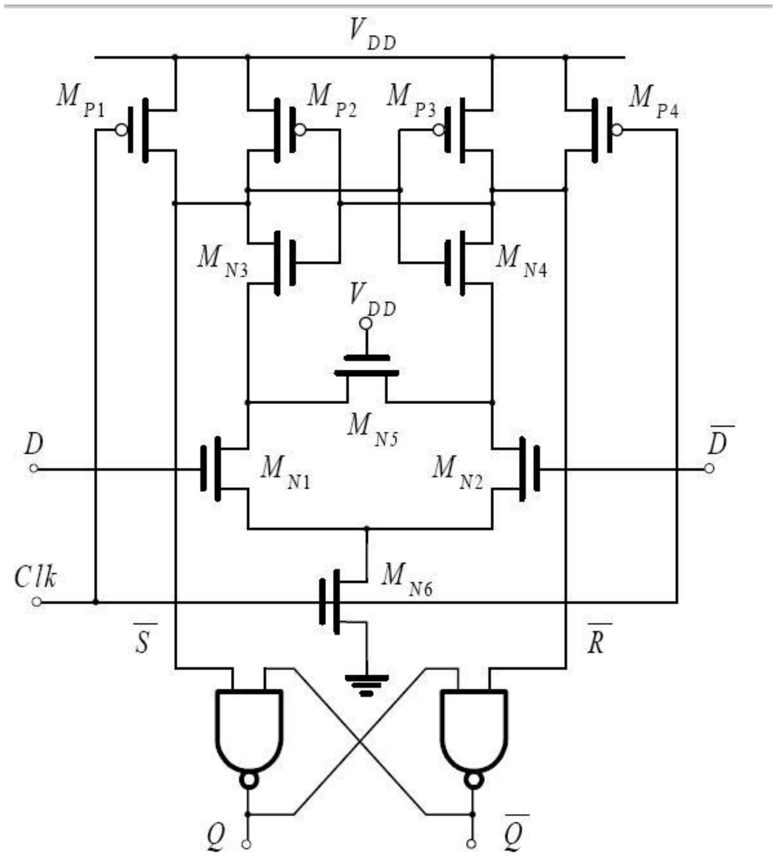


Figure 2.4: Sense amplifier based flip-flop [38, 28]

Fig. 2.5 shows a transmission gate based Master-Slave flip-flop that is a combination of two level-sensitive latches and is used in this thesis as the main circuit for investigating the effect of process variation on its timing yield. The advantages of this flip-flop are simplicity and excellent race immunity [30].

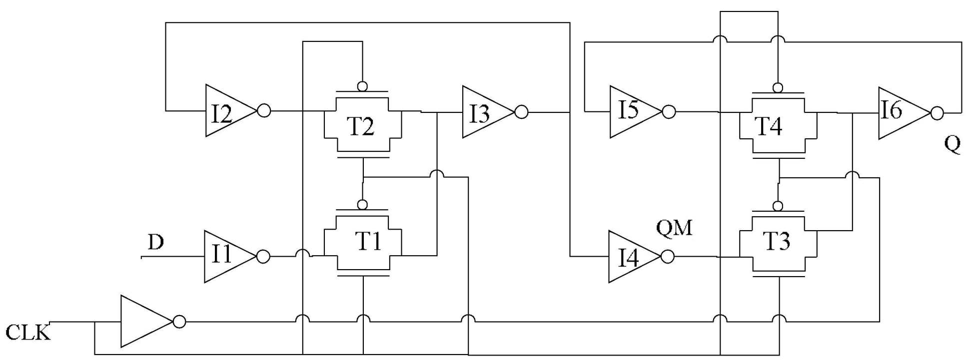


Figure 2.5: Transmission gate based Master-Slave flip-flop

Chapter 3

A High Performance Constant Delay Serial Adder

3.1 Introduction

In this chapter a new method for the addition of two operands in a serial addition (using a clock signal) is proposed. Because the operation of addition is used more than any other operation in a microprocessor and digital circuits, and high latency is the drawback of serial addition techniques, there is a need for efforts for gaining high performance serial adders. In serial addition the new method is independent of bit-widths of the operands and can perform addition in a constant delay. It is note-worthy that the Constant-Delay (CD) method significantly decreases the adder's hardware complexity. This is because the number of transistors has a linear relationship with the lengths of the words.

The rest of this chapter is organized as follows: The proposed structure for the addition of two operands with a constant delay is presented in section 3.2. In section 3.3, a comparison of the proposed CD adder with the conventional

Kogge-Stone adder in serial addition is probed. In section 3.4 and 3.5 the results and conclusions are presented.

3.2 Proposed adder

3.2.1 Main idea

As it was mentioned in the background section, the problems of the Kogge-Stone (KS) method is its hardware complexity, large area and power consumption. Suppose that two n-bit numbers, $A_{0...n-1}$ and $B_{0...n-1}$ are going to be added. Assume each pair of A_i and B_i is a column. For the addition of each column we have at most one carry that is sent to the next column. If a way can be found to stop propagating the carry to the next column the operation of addition can be performed faster. The main idea of the proposed adder is to use another digit, -1, in addition to the conventional {0, 1} digits for the representation of the binary numbers. Therefore, the proposal in this thesis is to represent each binary number using three digits {1, 0, -1}($3d$ form). For representing an n-bit $3d$ form number, two n-bit conventional form($2d$ form) numbers are used in this proposal. All of the bits of the original number (except bits with the value of -1) create the first n-bit number for the representaion of the $3d$ form number. The extra n-bit number is called Negative_Number. Each bit with the value of 1 in the Negative_Number demonstrates the existence of a bit with -1 value in the same position in the original $3d$ form number. The other bits in Negative_Number are 0. For example, if B=10-10011-110 and should be converted to the $3d$ form, two 10-digit numbers are appropriate. The first one is equal to 1000011010 and the extra one

(Negative_Number) is equal to 0010000100. Fig. 3.1 clearly shows the conversion from the $2d$ to the $3d$ form,

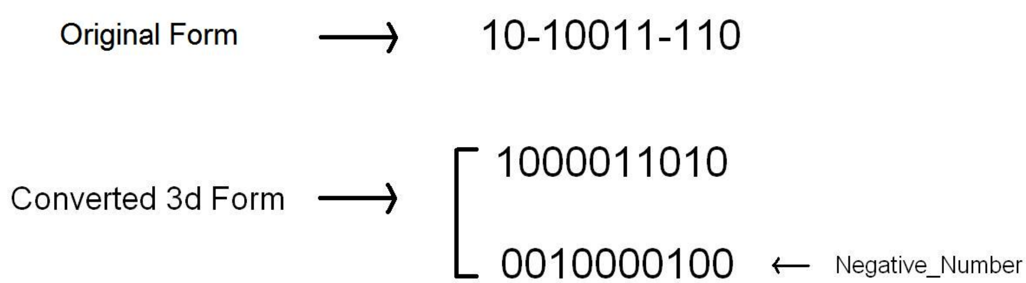


Figure 3.1: Conversion from $2d$ to $3d$ form

Now suppose that several numbers are going to be added in several clocks. In each clock two numbers are added. Using the proposed method which is shown in Fig. 3.2, in the first clock two $2d$ form numbers are added and the result of first clock is in $3d$ form. This addition requires a special adder which is described in details in section 3.2.2. In the other clocks one $2d$ form number is added with a $3d$ form number and the result is in $3d$ form and it requires another adder that is described in details in section 3.2.3. In the last clock, there is a $3d$ form result that should be converted to the $2d$ form. Here, there is a need for the final converter which is described in section 3.2.4 in details.

3.2.2 Addition of the $2d$ form numbers using $3d$ form representation

For the addition of two $2d$ form operands in the first clock, the bits are decomposed to 1-bit sets or columns and the addition of them is startedin parallel.

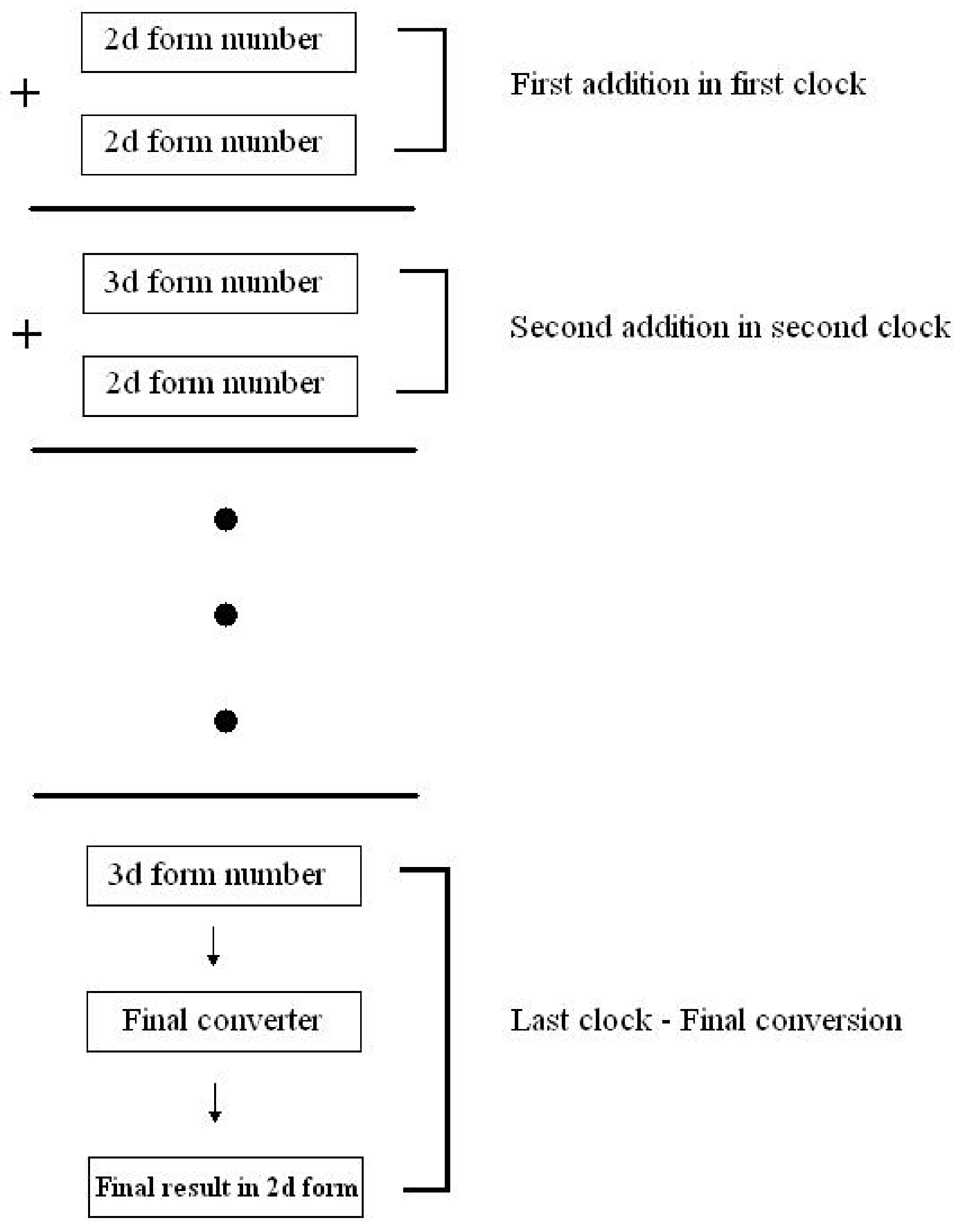


Figure 3.2: Addition of several numbers in several clocks using proposed method

This procedure can be seen in Fig. 3.3.

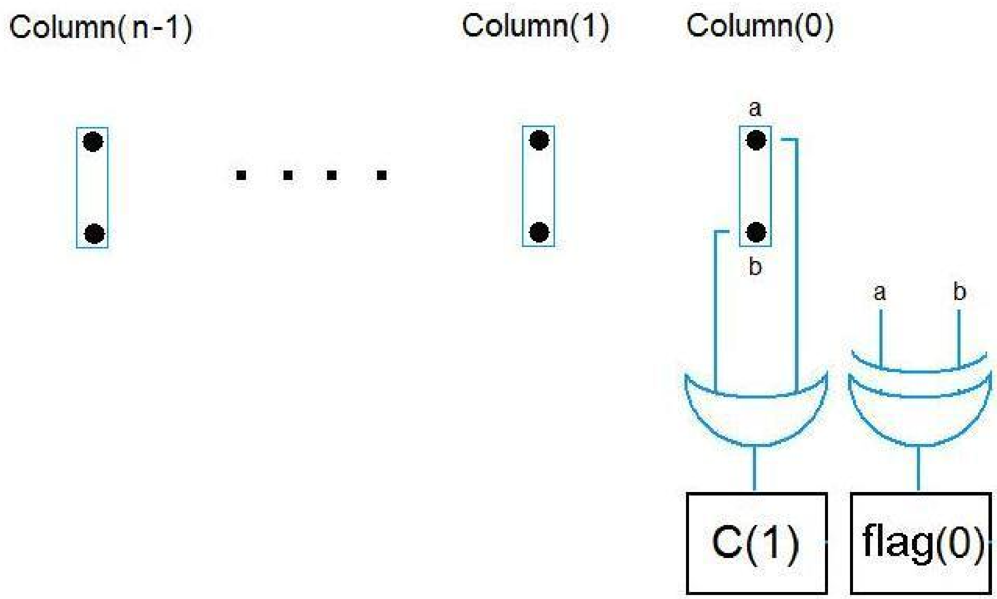


Figure 3.3: Addition of two $2d$ form numbers

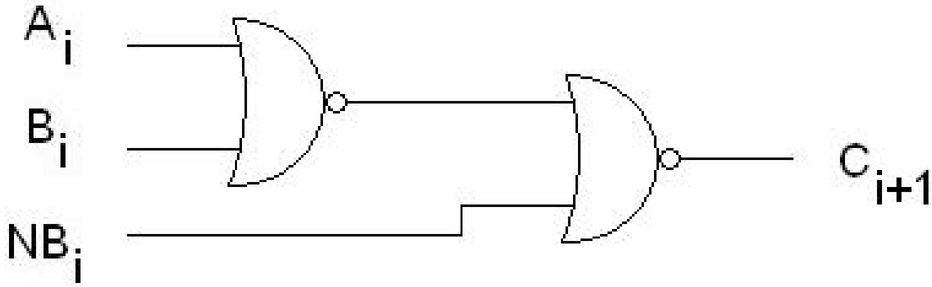
In order to prevent the carry propagation to the next columns, the cases, resulting in the propagation of the carries to the other columns should be found. Therefore, there are two cases: the first case is that both of the bits of one column are 1 and second case is that one of the bits of one column is 1. A Flag bit for each column is used to illustrate the condition of the addition regarding these cases. Therefore, an AND gate and a XOR gate need to be used as shown in Fig. 3.3. If the result of the 2-input AND gate is 1, a carry is propagated to the next column. If the second case occurs, i.e., one of the bits of one of the columns is 1, it's assumed that the anticipated sum result for that column is 1 and therefore a carry is given to the next column. In this case, a 1 in the corresponding Flag of that column is saved. A Flag with the value of 1 illustrates that the result of the addition of the current column is equal to one value greater than the real one. Therefore, to obtain the real result, 1 should be subtracted from the denoted result. In the next step, if both the input carries to the column and the Flag of that column are 1, i.e. the AND result of them is 1, the result should be zero. It means that instead of adding the input carry to the column with the result of the addition of that column which itself is one unit greater than the real result, nothing is done and the result is correct. If the input carry to the column is zero, while the Flag of that column is 1, the result should be -1. In this way, no carry is propagated to subsequent columns.

Fig. 3.4 shows a gate level implementation of C_{i+1} and $Flag_i$, which is used in this thesis.

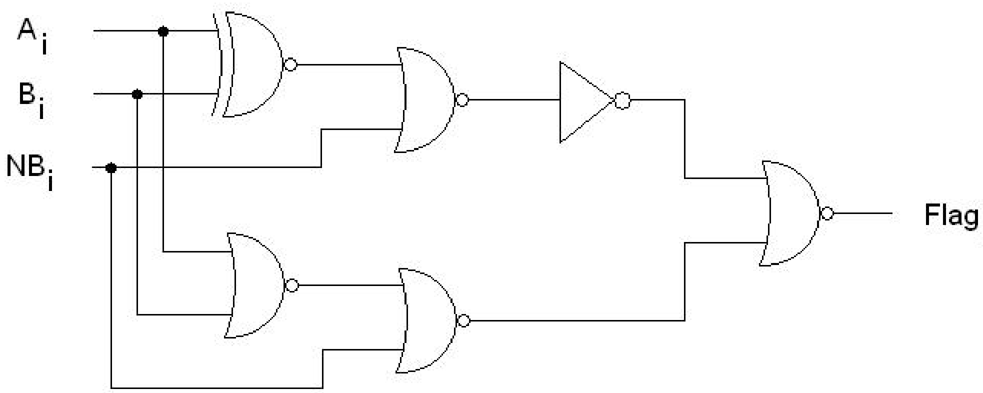
Now, two bits as the result bits are used, Sum and Negative_Sum(NSum). The Negative_Sum(NSum) number is 1 where the Flag value is 1 and previous column consists of two 0 bits. Because only in this case, which Flag is 1 and previous column bits are 0, does the wrong carry 1 propagate to the next stages and a subtraction is needed using NSum. For other Flag bits equal to 1, the right carry (1) is propagated. The truth table of the previous conditions is listed in Table. 3.1.

$$\Rightarrow Sum_i = \overline{Flag}_i \cdot C_i, \tag{3.1}$$

By computing :



(a)



(b)

Figure 3.4: (a) Gate level implementation of C_{i+1} in the CD adder and (b) Gate level implementation of $Flag_i$ in the CD adder. Note NB_i is the i_{th} bit of the Negative_Number.

$$\begin{aligned} NSum_i &= Flag_i \cdot \overline{A_{i-1}} \cdot \overline{B_{i-1}} \\ &= (A_i \oplus B_i) \cdot \overline{A_{i-1}} \cdot \overline{B_{i-1}}. \end{aligned} \tag{3.2}$$

Fig. 3.5 shows an example of the addition of two $2d$ form numbers (A=101101 and

Table 3.1: Producing Sum for $2d$ form input

$Flag_i$	C_i	Sum_i
0	0	0
0	1	1
1	1	0
1	0	0

B=10011) using $3d$ form representation.

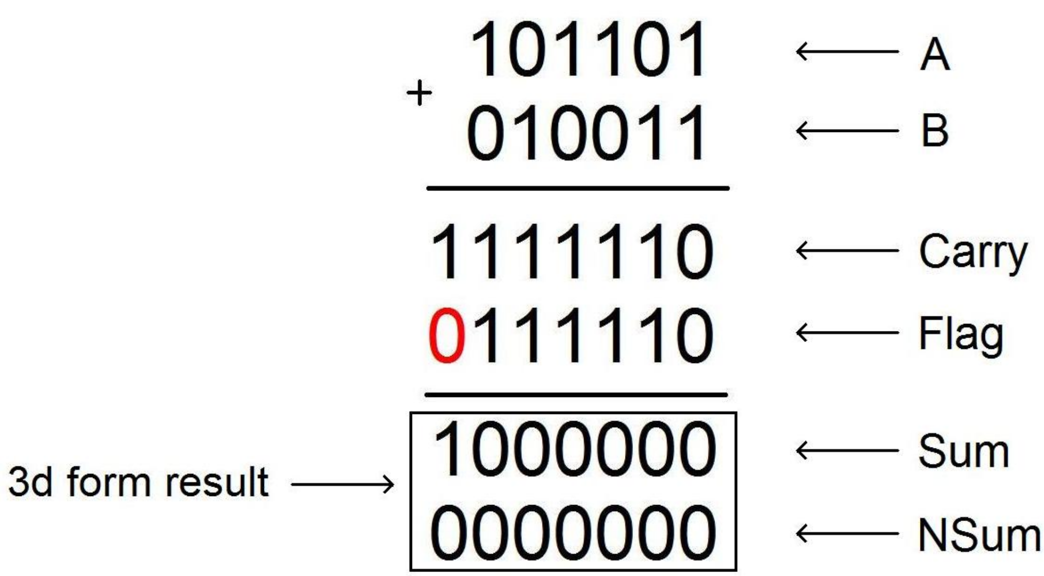


Figure 3.5: Addition of A=101101 and B=10011 using $3d$ form representation.

3.2.3 Addition of the $3d$ form numbers

This addition operation is similar to the one in the previous section, but one of the two inputs are in the $3d$ form and the other is in the $2d$ form. Hence, for the serial addition, in the middle of additions, there are two numbers to be added which the first one is in the $2d$ form and the other is the result of the calculation of the previous additions that is in the $3d$ form. Theorem1 shows that the result of the addition of one $2d$ number with one $3d$ number is in the $3d$ form.

Theorem1: the result of the addition of one $2d$ number (A) with one $3d$ number (B) is in the $3d$ form.

Proof: For each pair of digits, one digit of A and one digit of B, should be added and there are six possible choices: (0,0), (0,1), (0,-1), (1,0), (1,1), and (1,-1). The result of the addition of all of these choices are in this set {-1, 0, 1, 2} and no results such as -2 exist such that the result is in the $3d$ form.

Consider that in the middle of the serial addition, where two numbers are added, one in the $2d$ form and the other in the $3d$ form. This addition is similar to the addition of two numbers in the $2d$ form, we can do addition in $3d$ form but this small difference: another input, Negative_B($NB_{0 \ldots n-1}$) is included which is the extra n-bit number of the $3d$ form number.

Table 3.2: Producing Carry and Flag

NB_i	A_i	B_i	C_{i+1}	$Flag_i$
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	1	0	0	0

Table 3.2 and Table 3.3 is the summary of the addition operation in the $3d$ form.

Table 3.3: Producing Sum and NSum

Flag$_i$	C$_i$	Sum$_i$	NSum$_i$
0	0	0	0
0	1	1	0
1	1	0	0
1	0	0	1

The relation between the inputs and the outputs are expressed as follows:

$$\begin{aligned} C_{i+1} &= (A_i + B_i) \cdot \overline{NB_i} \\ &\Rightarrow \; C_i = \overline{(\overline{A_{i-1} + B_{i-1}}) + NB_{i-1}}, \end{aligned} \tag{3.3}$$

$$\begin{aligned} Flag_i &= (A_i \oplus B_i) \cdot \overline{NB_i} + \overline{A_i} \cdot \overline{B_i} \cdot NB_i \\ &\Rightarrow \; Flag_i = \overline{((\overline{A_i \oplus B_i}) + NB_i) \cdot \overline{((\overline{A_i + B_i}) \cdot NB_i)}}, \end{aligned} \tag{3.4}$$

$$\begin{aligned} Sum_i &= \overline{Flag_i} \cdot C_i \\ &\Rightarrow \; Sum_i = \overline{Flag_i + \overline{C_i}}, \end{aligned} \tag{3.5}$$

and

$$\begin{aligned} NSum_i &= \overline{C_i} \cdot Flag_i \\ &\Rightarrow \; NSum_i = \overline{\overline{Flag_i} + C_i}. \end{aligned} \tag{3.6}$$

In fact, there is no extra conversion process for converting from $2d$ to $3d$ form and vice versa in several clocks of serial addition except the final operation during the last clock of serial addition in which the $3d$ number result should be converted into a regular $2d$ form that is explained in details in the next section. So there is no power consumption for conversion in several clocks but there is power consumption for the adder that is used in final operation as a converter that is calculated in section 3.4. In the first clock two numbers are added and by using the result of first clock in the next clock, without any conversion process, the result of second clock will be in $3d$ form and similarly for other clocks there is no need for conversion operation. More details about number of transistors and hardware comparisons is explained in section 3.3.2.

Fig. 3.6 shows one block of the proposed CD adder method that produces Sum_i and $NSum_i$ and Fig. 3.7 is a block diagram of an n-bit proposed adder. In Fig. 3.7, A_{-1} and B_{-1} and NB_{-1} represent the input carry to the adder so that if $A_{-1} = B_{-1} = 0$, the input carry is 0, and if one of A_{-1} or B_{-1} is equal to 0 and the other is equal to 1, the input carry is 1. In these cases NB_{-1} is equal to 0.

3.2.4 Final operation - converting to conventional form

Suppose there are m n-bit numbers to be added in a serial manner by using a clock signal. In each clock the proposed CD adder is used and the results of each clock is in the $3d$ form. In the final clock, when all of the m numbers are added, the result is still in $3d$ form and should be converted to the conventional form ($2d$). The result includes 2 numbers, Sum and NSum, that each has $n + \log_2(m)$ bits length because the result of addition of m n-bit numbers is less than

Figure 3.6: CD adder block for one output bit

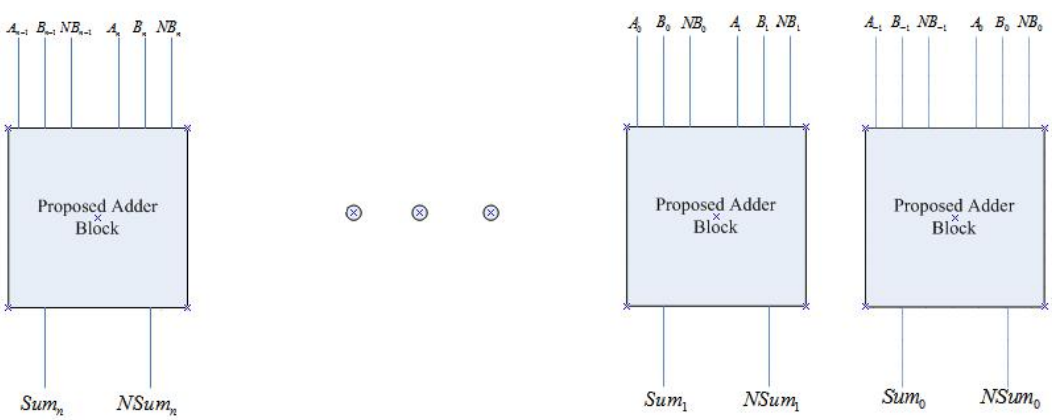


Figure 3.7: CD adder for the addition of two n-bit operands. The first block from the right side is used to produce carry in necessary cases.

$m \times 2^n = 2^{\log_2(m)+n}$. NSum conveys that in each position, its bits are 1, a subtraction needs to be done. Therefore, for converting to $2d$ form, NSum should be subtracted from Sum, which can be done by adding Sum by the 2's complement of NSum, in order to invert all the bits of NSum, and then adding the inverted NSum and also a carry equal to 1. Now the final stage contains an $n + \log_2(m)$ simple inverters and a $n + \log_2(m)$ bit adder. The final operation is not a constant delay procedure but it will be shown in section 3.3.1 that the whole adder in several clocks, if m and n as the number of operands and word length respectively, have some relation (that are met in many practical cases), will perform addition in a time less than a constant delay for every word length.

3.3 Comparison between CD and KS methods in serial addition

3.3.1 Delay comparison

For addition of m n-bit numbers, a $n + \log_2(m)$-bit adder is required. In the $n + \log_2(m)$ bit Kogge-Stone(KS) adder, $\log_2(n + \log_2(m))$ stages also exist on top of the first stage, producing a primary Sum, P, and G, where all have a delay equivalent to the XOR gate delay. Here, the last two stages produce the final carries and final sum that each have a delay equivalent to the XOR gate delay. Each of the $\log_2(n + \log_2(m))$ stages has a D operation. Such an operation combines two different P and G blocks. The delay of this block is equivalent to the delay of the AND gate. Also, the OR gate and the AND gate delays are supposed to be equal to (δ). For the CMOS implementation of the three gates, D operation has a delay equivalent to the XOR gate delay. As a result, the KS delay is

$$KS = XOR\_delay \times (\log_2^{n+\log_2^m} + 3). \tag{3.7}$$

According to (3.3), (3.4), (3.5), and (3.6), the CD adder exhibits a delay equal to

$$CD = XOR\_delay + 3 \times \delta. \tag{3.8}$$

To do addition of m n-bit numbers using the CD adder in several clocks with a Kogge-Stone adder as the final conversion circuit block, the total delay is expressed as

$$Total = (m-1) \times (XOR\_delay + 3 \times \delta) + XOR\_delay \times (\log_2^{n+\log_2^m} + 3)). \tag{3.9}$$

Thus, the delay of each addition by the CD adder in a clock, using the final KS adder, is equal to (3.9), divided by (m-1) and the following is attained:

$$Final\_CD = XOR\_delay + 3 \times \delta + XOR\_delay \times (\log_2^{n+\log_2^m} + 3) \Big/ m-1. \tag{3.10}$$

Then, assuming that the XOR delay is equal to 2δ such that

$$m = 2 \Rightarrow \begin{cases} KS = (2\log_2^{n+1} + 6) \times \delta \\ CD = (5 + (2\log_2^{n+1} + 6) \big/ 1) \times \delta \end{cases} \Rightarrow \forall n \geq 1 \rightarrow CD > KS \tag{3.11}$$

Ignoring which numbers to be added, Fig. 3.8 reflects the above equations as the delays for KS and proposed CD adder when m=2. It can be seen that if just 2 numbers are added serially, CD method is not faster than KS method, but if m is greater than or equal to 3, the proposed CD works faster than KS method for serial additions as it can be seen in Fig. 3.9.

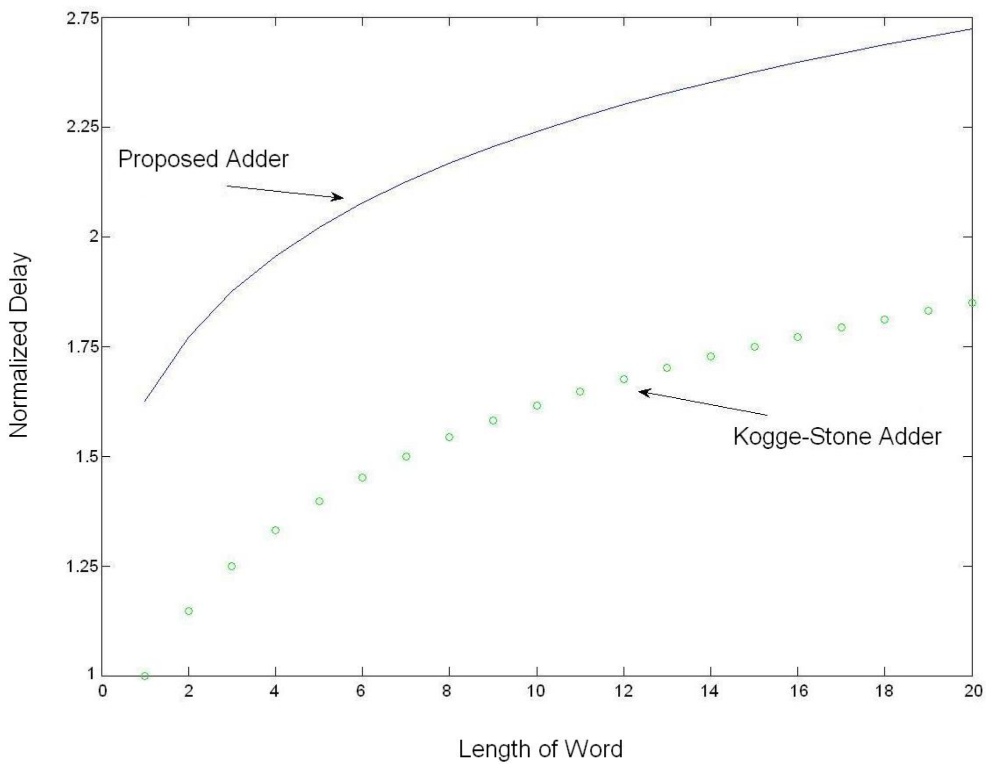


Figure 3.8: Comparison of the normalized delay of the KS and the proposed adder when m=2

The delay of KS and proposed for m=3 is computed by

$$m = 3 \Rightarrow \begin{cases} KS=(2\log_2^{n+1.58}+6)\times\delta \\ CD=(5+(2\log_2^{n+1.58}+6)/2)\times\delta \end{cases} \tag{3.12}$$

$$\Rightarrow \forall n \geq 2.42 \rightarrow CD < KS$$

$$m \geq 4 \Rightarrow (\forall n \geq 1 \rightarrow CD < KS).$$

Similar to fig. 3.8, ignoring which numbers to be added, fig. 3.9 reflects the above equations as the delays for KS and CD adder when m=3. Fig. 3.8 and 3.9 were selected to be shown because for m=2 and m=3, the CD adder has the worst scenario in terms of delay compared to KS adder and by increasing m, the CD adder performs faster and faster compared to KS adder.

So, if $m \geq 3$, almost in all the conditions($n \geq 3$), the proposed CD adder works faster than the KS adder. According to (3.7) and (3.10), if $(\log_2^{n+\log_2(m)}+3) \leq m-1 \Rightarrow n \leq 2^{m-4} - \log_2^m$, the $Final\_CD$ is a delay less than a constant value that is equal to $2XOR + 3\delta$. If $m \geq n+5$, $\forall n \geq 2$ the newly developed method has a delay less than the constant value of $2XOR + 3\delta$.

3.3.2 Area and power comparison

An $n + \log_2(m)$ bit KS adder has $\log_2(n + \log_2(m))$ stages [33]. In each stage, $O(n + \log_2^m)$ is the hardware(number of transistors) such that the total hardware in the KS adder is

$KS\_Hardware = O((n + \log_2^m) \times \log_2^{n+\log_2^m})$.

In the CD adder, the $n + \log_2^m$ blocks, like which is used in Fig. 3.6, are necessary

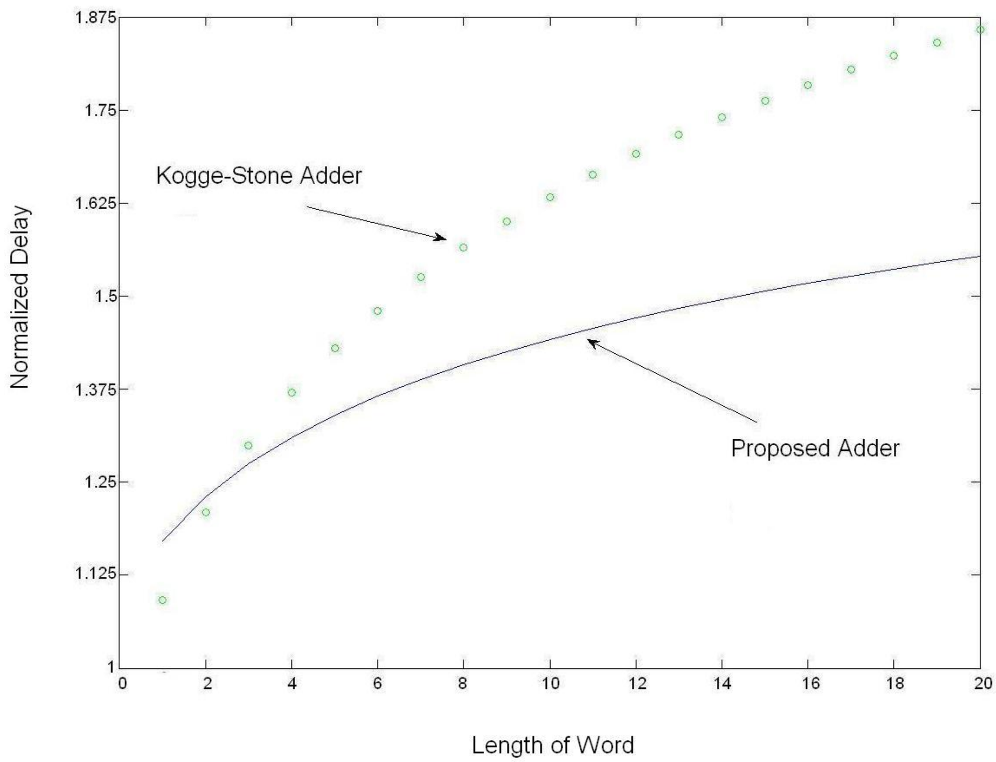


Figure 3.9: Comparison of the normalized delay of the KS and the proposed adder when m=3

and this block contains O(1) hardware. So, the total hardware of the proposed CD is calculated by

$CD\_Hardware = O(n + \log_2^m)$.

The proposed CD adder is not only superior in terms of the delay order, to the Kogge-Stone (KS), but also in terms of the hardware order and power. The proposed CD adder can be expanded for addition of larger numbers so easily because in each clock, CD adder blocks for different digits are independent and just by placing the exact copy of the block at the end of the two numbers, it can be expanded to (n+1)-bit adder. So, CD adder shows so much lower hardware complexity compared to KS adder.

3.3.3 Brief algorithmic comparison with other similar adders

In [31], similar concept of 0,1, and -1 digits is described. In [34], a redundant binary Multiplication-and-Accumulation (MAC) unit is proposed. In MAC implementation, a redundant binary multiplier and redundant binary adder has been used [35, 36]. The structure of the redundant binary multiplier and the redundant binary adder are similar to the proposed structure in [31]. The proposed adder in [31] is appropriate for fast multiplication and division but not in serial additions and multioperand additions. The problem with this method compared to the proposal in this thesis is that for representing a simple output equal to 1 in a special significance, another output bit equal to 1 has been used that has a higher significance and generates another problem [32]. The authors use two digits for representation of each output that are with different significances, like the full adder, and for each addition in the serial addition, the converter in [32] is needed to be used and the delay of a converter has a linear relationship with the word length. To reduce this delay, as it is explained in [31], a high number of transistors and a large area are required. This method in [31] works a little faster than the carry look-ahead adder and with a much larger area. It is note worthy that the proposed CD method produces two digits of output, each with the same significance, and does not produce any output for the more significant ones. It can be used in each serial addition (except the last addition) without conversion, and consequently works in a constant latency.

The comparison between the power dissipation of the proposed CD and Kogge-Stone (KS) techniques using Cadence simulator is discussed in the next section.

3.4 Experimental results

In this thesis, a serial adder that can add many numbers in several clocks is realized. A master-slave positive edge register is employed for keeping the output results of each clock. This register is composed of two positive level-sensitive latches and negative level-sensitive latches. The bulk of the transistors is connected to the source. The sizing of the transistors is done for both the KS and proposed CD method in the same way, i.e., the sizing for the worst-case delay and equivalent to the ratio of the PMOS and NMOS in an inverter for a pull-up and pull-down network (The width of the PMOS should be two times bigger than the NMOS in an inverter).

The simulation and implementation were done using Cadence software and schematic was used as circuit design style. The delay is calculated from 50 % of the input to 50 % of the output in 180nm technology and 1.8v supply voltage in a 100MHz clock. The new method is implemented by using the CMOS technique and the AND, OR, XOR, and NOT gates, according to (3.3), (3.4), (3.5) and (3.6). A comparison of these results reveals that the improvement ratio for the delay is

Table 3.4: Comparison of the serial addition of 64 58-bit numbers in one clock duration (addition of two 64-bit numbers) in the KS and CD methods with the consideration of the register

	KS	CD
Delay (ps)	1508.5	524.86
Average Power (mW)	6.96	3.93
Number of Transistors	8492	5330

65.2% and for the average power dissipation is 43.53%. This improvement makes sense according to (3.7) and (3.10) for n=58 and m=64, because proposed CD adder in each addition of serial additions works with a constant delay but KS works with a logarithmic delay. It was used 64 words each with 58-bit length because the result of addition of these numbers is a number with $'58 + \log_2(64) = 64'$ bit width and a 64-bit adder should be used to add them up in 64 clocks. The numbers that were added for KS adder are $2^58 - 1 = 288230376151711743$ in addition to 63 equal numbers that are $2^57 - 1 = 144115188075855871$. It was seen in simulation that addition of these numbers has the lowest delay for KS adder. Also for proposed CD adder, it was used 63 equal numbers that are $2^58 - 1 = 144115188075855871$ in addition to one $3d$ form number in the first clock (suppose (A1, NA1)), that both A1 and NA1 are equal to $2^58 - 1 = 144115188075855871$. It was seen in simulation that the addition of these numbers has the highest delay for the proposed CD adder, compared to the addition of other numbers. So comparing the lowest delay for KS adder and the highest delay for CD adder, with the same word with, results the worst case scenario of improvement.

3.5 Conclusion

In this chapter, a new algorithm for serial addition and comparison of the performance and power dissipation of the proposed method with other high performance adders in serial additions are developed and discussed. The proposed adder, CD adder, is not only better than Kogge-Stone (KS) method in terms of latency but also is better in terms of power dissipation in serial additions. Also, if the number of operands becomes a little more than the word length, the proposed method performs addition in a constant latency.

In the next chapter, the statistical design of a Master-Slave D flip-flop will be investigated in details to gain maximum timing yield for setup-time and hold-time under process variations.

Chapter 4

Framework for Statistical Design of a Flip-Flop

4.1 Introduction

The uncertainty of gate delays due to process variation causes considerable uncertainty in the performance metrics of flip-flops. Because flip-flops are designed small, process variation can cause significant effects in the timing yield of them. In this thesis, it's assumed that the widths of transistors in flip-flops have variations and the effect of the widths of transistors is taken into account.

In order to assess how different metrics of flip flop performance are sensitive to process variation, the behavior of the flip-flop shown in fig. 4.1 is simulated by SPICE Monte Carlo simulation. Fig. 4.2 shows how the signal propagation delay from input D to node QM (TDQM) varies due to variation of the widths of transistors. Consequently, setup-time, which is limited by TDQM, becomes uncertain and results in timing violations. Further to setup-time, as shown in fig. 4.3, clock-to-output (Clock-to-Q) delay becomes uncertain and impacts on the design operational frequency.

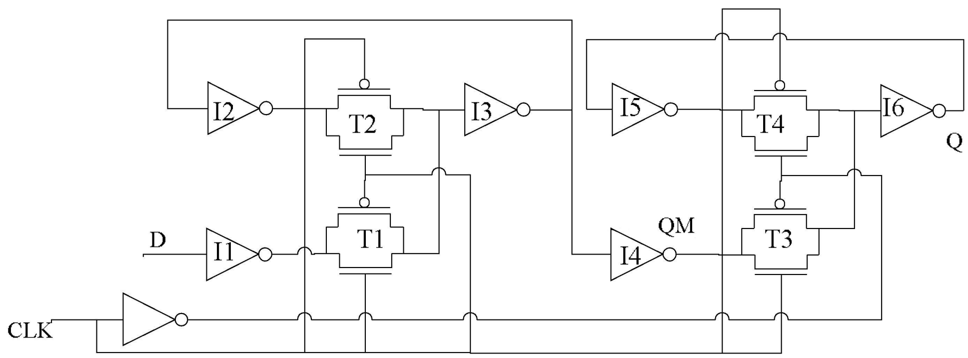


Figure 4.1: Master-Slave D flip-flop

Uncertainty in flip-flop's characteristics can significantly decline timing yield because around 10 % of all design components are flip-flops and they directly impact on operational frequency. Therefore, it is crucial to desin robust flip-flops against process variation.

In this chapter statistical design of a Master-Slave flip-flop using a yield maximization method is investigated and valuable results have been found. The objective of this chapter is to propose a method to increase the flip-flop robustness against variations in the widths of transistors. To achieve this goal, it is attempted to determine the flip flop's gate sizes considering performance and area constraints.

4.2 Basic concepts

A Master-Slave flip-flop architecture, shown in Fig. 4.1, is chosen as the case study in this thesis. For the rest of the chapter, the equations and performance

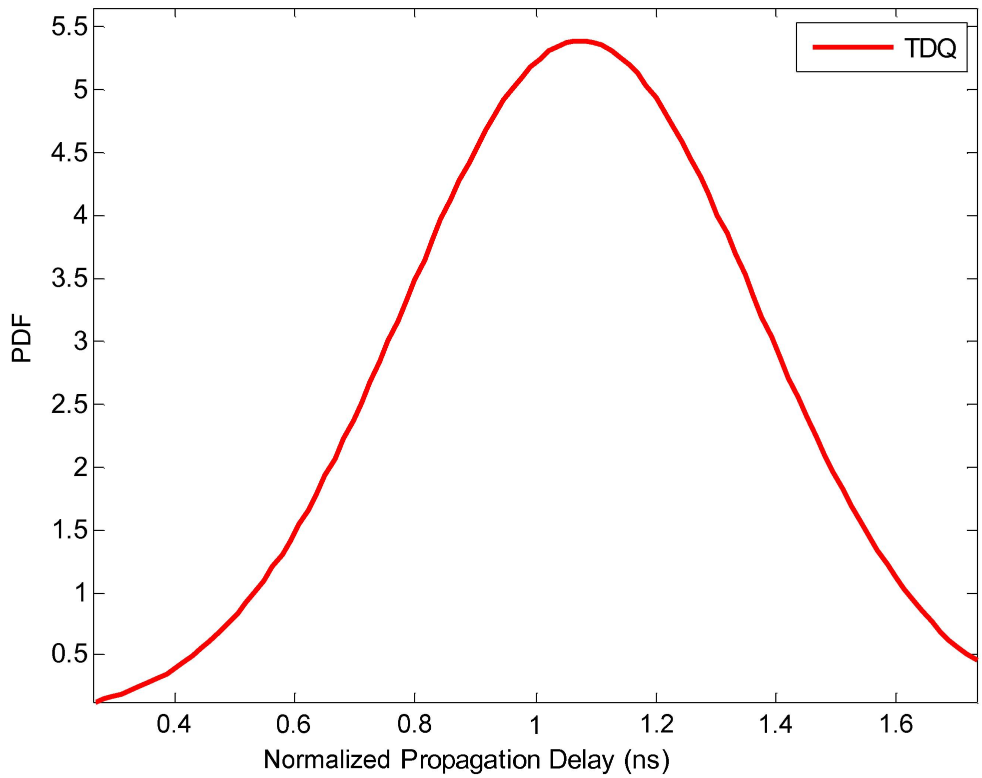


Figure 4.2: Variability of D-to-Q delay of flip-flop, $\sigma = 807.3ps$

metrics are expressed in relation to the selected architecture and the effect of variations on the widths of transistors in timing yield is studied.

Conventionally, the minimum clock period of a design (T_{clk}) is restricted by the following constraints:

$$\forall(i,j) \in subsequent\ flip-flops\ of\ design:$$
$$\begin{cases} skew_i - skew_j + D_{ij}^{\max} \leq T_{clk} - T_{setup} \\ skew_j - skew_i + T_{hold} \leq D_{ij}^{\min} \end{cases} \tag{4.1}$$

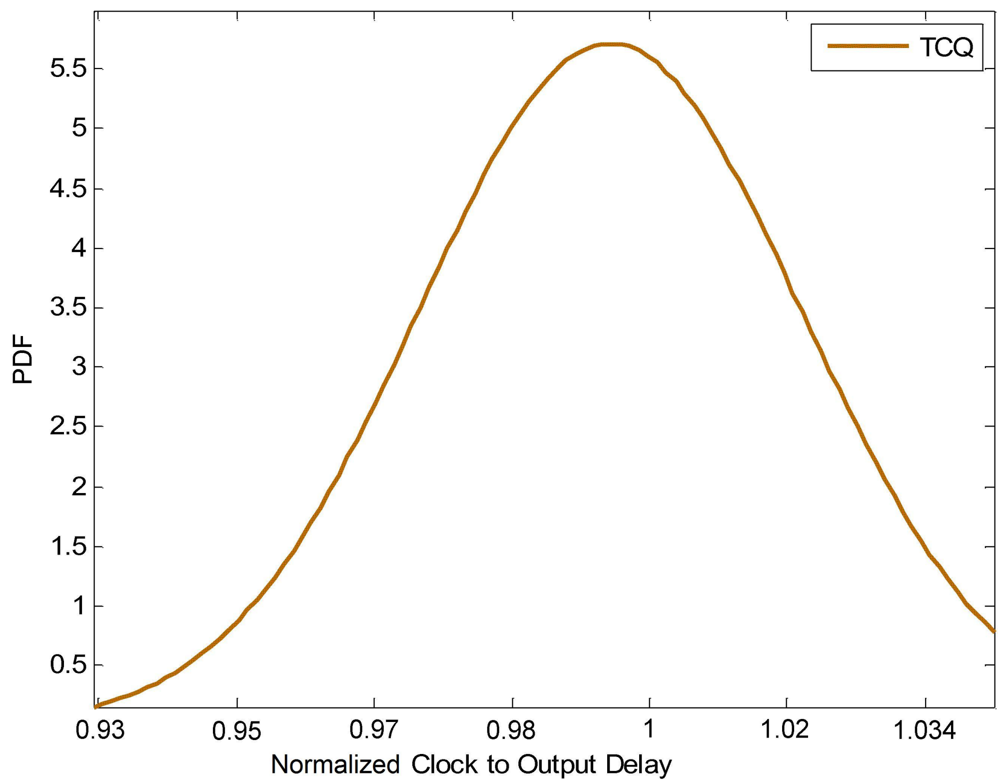


Figure 4.3: Variability of Clock-to-Q delay, $\sigma = 430.2ps$

where $skew_i$ and $skew_j$ are the clock arrival times of the flip-flops and $D_{ij}^{\max}$ and $D_{ij}^{\min}$ are the maximum and minimum combinational delays between subsequent flip-flops. In above Equations, T_{setup} and T_{hold} represent setup-time and hold-time of flip-flop which are considered as the timing parameters of flip-flops.

The setup-time is defined as the latest allowed data arrival with respect to the clock edge in order to correctly capture the data. The hold-time is defined as the earliest time after the clock edge is triggered, at which the input is allowed to be changed. An example is shown in Fig. 4.4.

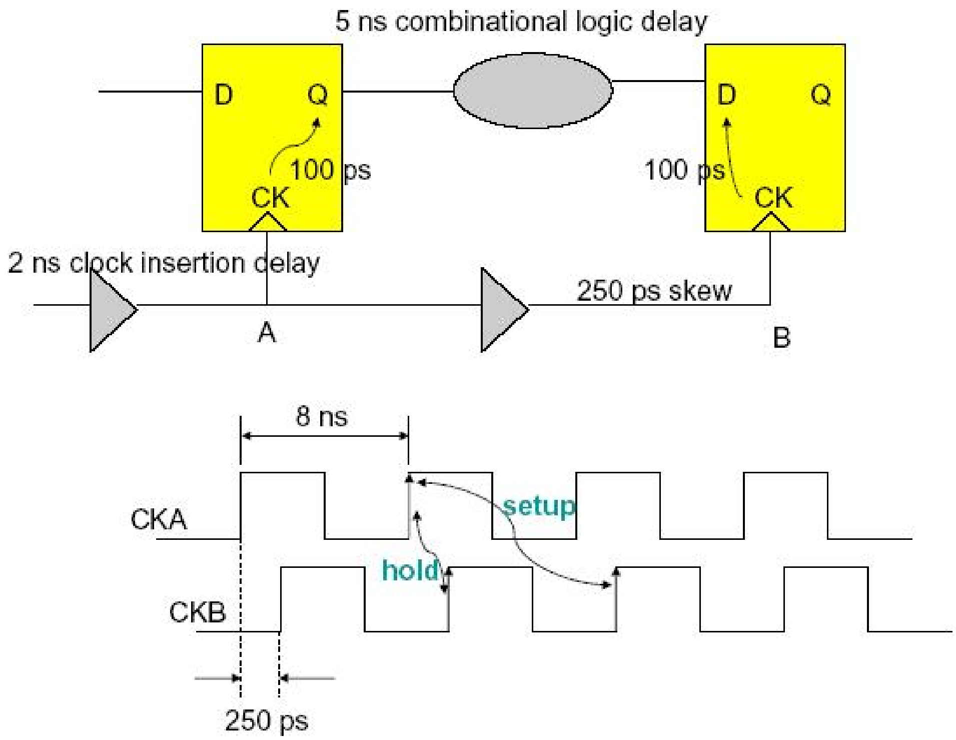


Figure 4.4: Setup and Hold time illustration

According to Fig. 4.1, both setup-time and hold-time can be specified in terms of the delay of different components of flip flop as:

$$T_{setup} = 3 \times Delay_{inv} + Delay_{tgate} \tag{4.2}$$

where $Delay_{inv}$ and $Delay_{tgate}$ represent propagation delay of inverter and transmission gate, respectively.

For analyzing the design constraints that will be discussed in the next section, three parameters that are functions of some constant parameters are considered.

1. Maximum area as a function of the widths of transistors (Area-Max)

2. Setup-time (T_s)

3. Delay of flip-flop (T_d)

Now the task is to find these parameters. For achieving a better delay, the gate sizing should occur on the pass-transistor and inverter gates. Therefore, it is assumed that the width of a PMOS transistor in both the pass-transistor and inverter gates are twice the width of an NMOS for the "critical path" sizing [38]. Here, the new parameter, the width of the gate, is defined. The width of the gate is assumed to be equal to the width of the NMOS (W_n), simplifying the calculations ($W_p = 2 \times W_n$). In the rest of the chapter W_i stands for the width of inverter gate and W_t represents the width of the pass-transistor gate in the circuit of Fig. 4.1. It's assumed that the maximum area (Area-Max) is the summation of the widths of all of the gates. Area-Max, setup-time and the delay of the flip-flop are obtained as follows:

1. Area-Max $= \sum W_i + \sum W_t$

2. $T_s = T_{inv1} + T_{inv3} + T_{inv4} + T_{pass1}$

3. $T_d = T_{inv6} + T_{pass3}$

Thus, the delays of all of the gates must be found. Because deep sub-micron technology (45nm) is used in this thesis, it's assumed that the operational region of the transistors in this flip-flop is velocity saturation. For each pass-transistor and inverter, according to the calculations in [38],

$$T_{pass(j)} = 0.69 \times (R_{eq-pass(j)}) \times (C_{o-pass(j)}) \tag{4.3}$$

$$and$$

$$T_{inv(i)} = 0.69 \times R_{eq-inv(i)} \times C_{o-inv(i)}$$

$$for\ (1 < i < 6)\ and\ (1 < j < 4),$$

where, $R_{\mathrm{eq-pass(j)}}$ is the equivalent resistance of the pass-transistor and $C_{\mathrm{o-pass(j)}}$ is the output capacitance of the pass-transistor [38]. Also,

$$C_{in-inv(i)} = 2 \times C_{ox} \times L_{eff} \times W_i, \tag{4.4}$$

$$C_{in-pass(j)} = 2 \times (C_j \times L_s \times W_t) + C_{jsw} \times (C_j \times L_s + W_t)$$

$$for\ (1 < i < 6)\ and\ (1 < j < 4).$$

where, C_{ox} is the capacitance of the gate oxide of the transistors, L_{eff} is the effective channel length of the transistors, C_j is the zero-bias bottom bulk capacitance of the transistors, C_{jsw} is the zero-bias sidewalk bulk capacitance of the transistors and L_s is the sidewalk length of the transistors. According to Fig. 4.1, the following equations for the input and output capacitance of the inverters and pass-transistors is obtained. The input (or output) capacitance of a pass-transistor or an inverter is calculated by using the capacitance of all the gates, connected to their input (or output) node. The following equations are required to find the setup-time and the delay the flip-flop (assumption is transistors are working in the velocity saturation operational region):

$$I_{ds(n/p)} = ((V_{dd} - V_{T0}) \times V_{DSAT(n/p)} - V_{DSAT(n/p)}^2/2) \times k_{n/p}/(L_{eff} \times W_i), \quad (4.5)$$

$$R_{eq(n/p)} = 0.75 \times V_{dd} \times (1 - 7/9 \times \lambda_{(n/p)} \times V_{dd}),$$

$$and$$

$$R_{eq-inv} = (R_{eqn}/Idsn + R_{eqp}/I_{dsp})/2, [38]$$

where $I_{ds(n/p)}$ is the drain source current of the NMOS or PMOS, $R_{eq(n/p)}$ is the equivalent resistance of the NMOS or PMOS, and R_{eq-inv} is the total resistance of the inverter. By using equations, T_{inv} is found. Also, by using the following equations, T_{pass} can be obtained [38]:

$$R(n\ or\ p) = L_{eff}/(k_{(n/p)} \times W_t \times (V_{dd} - V_{T0})) \quad (4.6)$$

$$and$$

$$R_{eq-pass} = R_n \times R_p/(R_n + R_p),$$

where, $R(n\ or\ p)$ is the resistance of a PMOS or NMOS in the pass-transistor, and $R_{eq-pass}$ is the total resistance of the pass-transistor. By finding T_{inv} and T_{pass}, the setup-time and the delay of the flip-flop can be calculated. Consequently, all of the deterministic problems can be analyzed.

4.3 Problem formulation

The problem here is finding the best values as the widths of gates of a Master-Slave flip-flop, which is shown in Fig. 4.1, to gaining the maximum timing yield under the gate width variations.

4.3.1 Statistical design

Design parameters

In this thesis, according to Fig. 4.1 the design parameters are the widths of the gates that are $W_{i(j)}(1 < j < 6)$ for the inverters, and the $W_{t(i)}(1 < i < 4)$ for the pass-transistors, and these are random variables. This problem has ten design variables, a large number. Also, T_s and T_d are found using equations 4.4-4.7.

Design constraints

There are three constraints in this problem. The first one is about the Area-Max, the second one is about the setup-time, and the third one is about the delay of the flip-flop. The fourth and fifth constraints are used for considering a minimum size for the width of each of the gates. As a result, the following constraints are attained:

1. $\sum W_i + \sum W_t < Area$

2. $T_s < Setup - Time$

3. $T_d < Hold - Time$

4. $W_{i(j)} > Minimum\ Inverters\ Width$,$(1 < j < 6)$

5. $W_{t(i)} > Minimum\ Pass - transistor\ Width$,$(1 < i < 4)$

Above constraints are shown in Fig. 4.5. The constant values of Area, Setup-Time, Delay, Minimum Inverters Width, and Minimum Pass-transistor Width are shown in Table. 4.2. The next section is the discussion of the applied yield maximization method, proposed in this thesis.

4.3.2 Yield maximization

Polyhedral approximation of the constraint region

The usage of the original constraints for analyzing is difficult, because they are nonlinear. Consequently, the first step is approximating the original constraints by the linear constraints. In this step, $\partial(T_s)/\partial(W_{i(j)})$ for all $(1 < j < 6)$, $\partial(T_s)/\partial(W_{t(i)})$ for all $(1 < i < 4)$, $\partial(T_d)/\partial(W_{i(j)})$ for all $(1 < j < 6)$, and $\partial(T_d)/\partial(W_{t(i)})$ for all $(1 < i < 4)$ should be taken for the approximation of the original constraints to allow the use of the Taylor series. For the Taylor series approximation, partial derivatives of all the variables should be taken. By using random vector x in the feasible region, the original constraints can be approximated by [39]-[40] computing

$$h_i(x) \approx h_i(x^*) + g_i(x^*)^T(x - x^*), \tag{4.7}$$

where $g_i(x^*)$ is the gradient vector of h, and h_i is the i_{th} original constraint. At first glance, point x^* is on the surface $h_i(x) = 0$ and has the minimal distance from x^c, which is the center of the initial tolerance box and is shown in Fig. 4.5. For finding the best match for the approximated linear constraints, an optimization problem should be solved [39]-[40] as follows:

$$\min \beta = [(x - x^c)^T (x - x^c)]^{\frac{1}{2}}, \tag{4.8}$$

$$subject \quad to \qquad h_i(x) = 0.$$

By solving this optimization problem, some linear constraints are found to solve the yield maximization problem instead of using the original ones.

Modeling arbitrary distributions

For traditional designs, it is assumed that symmetrical distributions simplify the solution process. In this case, the maximum volume box corresponds to the maximum yield that can be attained which is shown in Fig. 4.5. But, if Probability Density Function (PDF) is nonsymmetrical, the maximum volume box does not correspond to the maximum yield. Thus, the calculation of the yield involves the evaluation of a multi-dimensional probability integral by quadrature or Monte-Carlo-based methods, which is computationally expensive [41]. Here, Kumaraswamy's distribution [42] is used for approximating a Double-Bounded Probability Density Function (DB-PDF) for physically bounded variables such that

$$f(z) = abz^{a-1}(1 - z^a)^{b-1} \tag{4.9}$$

$$and$$

$$z = \frac{x - x^{\min}}{x^{\max} - x^{\min}} \quad , \qquad x^{\min} \leq x \leq x^{\max}.$$

where, $x^{\min}$ and $x^{\max}$ are the maximum and minimum possible values for the probabilistic variable x. DB-PDF can take different shape by using different values for 'a' and 'b'. The integral of the DB-PDF; i.e., CDF (Cumulative Density Function), can be calculated according to [42]

$$F(z) = 1 - (1 - z^a)^b. \tag{4.10}$$

Yield maximization

For the given W_i^{max}, W_i^{min}, W_t^{max} and W_t^{min}, and the nominal design that is the center of the tolerance box, the tolerance box can be found from the following representations:

$$T_{i(j)} = W_{i(j)}^{max} - W_{i(j)}^{min}(1 < j < 6) \tag{4.11}$$

$$and$$

$$T_{t(i)} = W_{t(i)}^{max} - W_{t(i)}^{min} (1 < i < 4),$$

where $T_{i(j)}$ is the tolerance box width for the i_{th} inverter, and $T_{t(j)}$ is the tolerance box width for the j_{th} pass-transistor.

If W_i^l and W_i^u define the bottom left and top right corner of the yield box that is the smaller box in Fig. 4.5, the yield is calculated by [39]-[40]

$$Yield(x^r, x^l, x^u) = \prod_{j=1}^{n} \Pr\{x_j^l \leq x_j \leq x_j^u\} \tag{4.12}$$

$$and$$

$$= \prod_{j=1}^{n} \left[F\left(\frac{x_j^u - x_j^r}{t_j}\right) - F\left(\frac{x_j^l - x_j^r}{t_j}\right) \right],$$

where x refers to W_i and W_t, and x^r refers to the bottom left corner of the tolerance box. Also F(x) can be found by using the integral of Kumaraswamy's distribution. Now, with a given tolerance box, the objective is to move this box such that the yield is maximized. Finally, the optimization problem is [39]-[40],

$$\max \quad Yield(x^r, x^l, x^u), \tag{4.13}$$

$$subject \quad to$$

$$A^+ x^u - A^- x^l \leq C,$$

$$x^r \geq x^{\min},$$

$$x^l \geq x^r,$$

$$x^u - x^l \geq t,$$

$$and$$

$$x^r + t \leq x^{\max}.$$

where x refers to W_i and W_t, and x^r refers to the bottom left corner of the tolerance box. A_i, as the transpose of the gradient vector (g in equation 4.7), can be calculated by the linearization of the performance constraint (h_i) at a given vector x. A^+ and A^- are the upper and lower bounds of the same performance constraint, and C refers to the constant terms in the linearization. Fig. 4.5 illustrates the ideas that have been discussed in this thesis.
By solving the above optimization problem using Matlab, maximum yield can be found.

4.4 Experimental results

In this thesis, 45 nanometer technology for the statistical design of a flip-flop is chosen. The feasible region in this problem is a ten-dimensional space that cannot be plotted. Therefore, a two dimensional feasible region is drawn in Fig. 4.6. This work can be easily extended to other technologies and other circuits. Table 4.1 lists the constant parameters of this technology [37]. The previously discussed

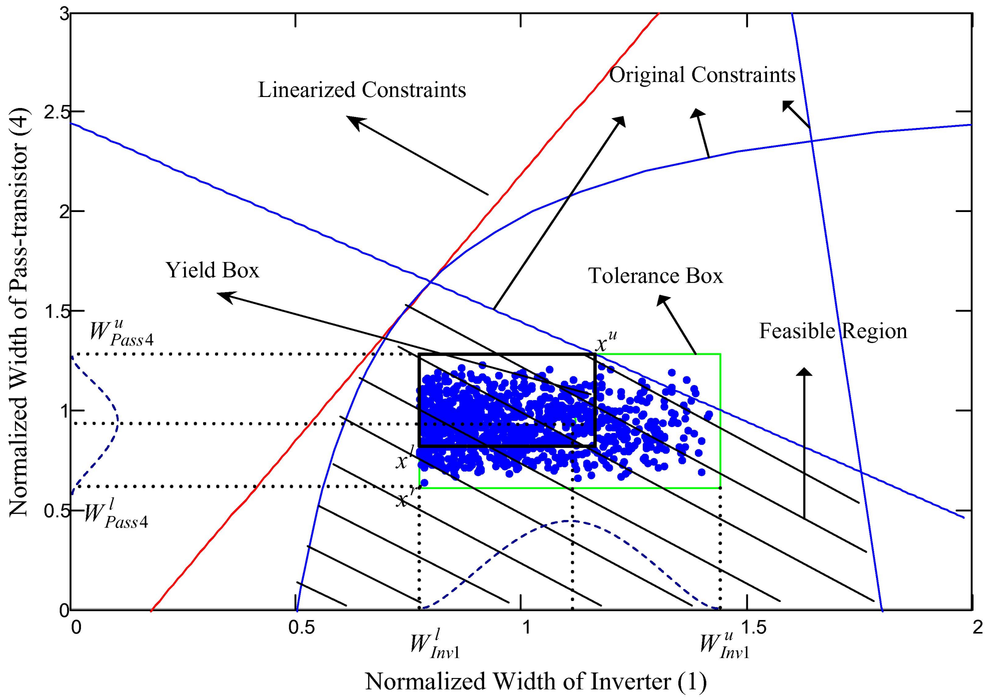


Figure 4.5: Normalized simplified yield maximization method

timing yield maximization method is implemented in Matlab.

Table 4.1: Constant parameters of 45nm technology

Parameter	Value	Parameter	Value
L_s	80×10^{-9}	μ_n	350×10^{-4}
C_j	5×10^{-4}	μ_p	100×10^{-4}
C_{jsw}	5×10^{-10}	L_{eff}	17.5×10^{-9}
λ_n	0.1	V_{T0}	0.18
λ_p	0.2	V_{DSATn}	33.23×10^{-3}
ϵ_{ox}	1.4×10^{-12}	V_{DSATp}	94.15×10^{-3}
T_{ox}	1.1×10^{-9}	V_{dd}	1

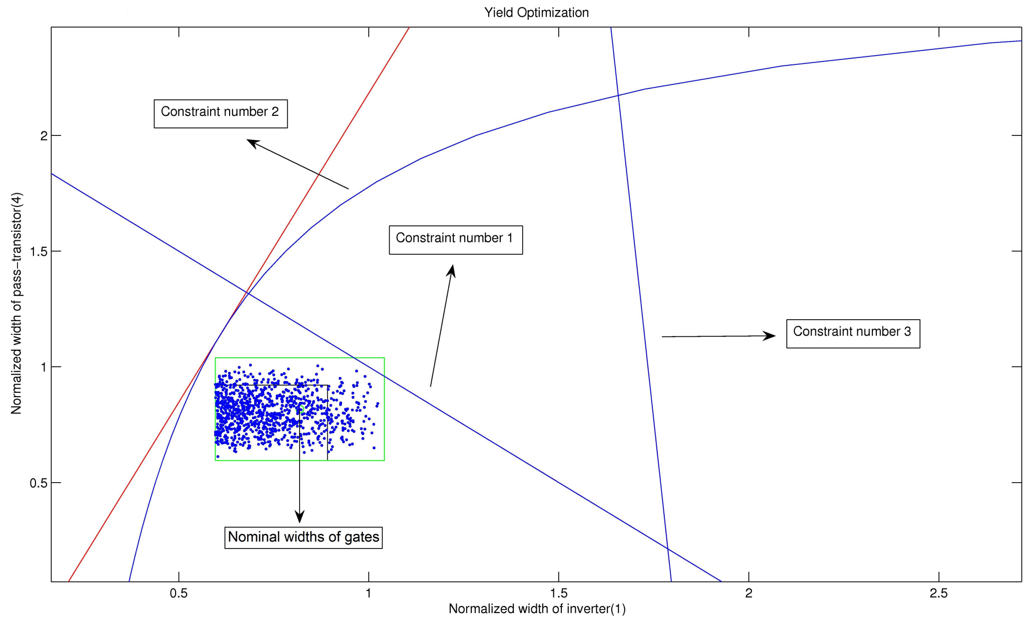


Figure 4.6: Yield maximization method for T=12 and Area=500×10^{-9}

For the design constraints in Table 4.2, the details about the experimental results for different tolerance boxes and different values as Area are summarized in Table 4.3. Design variables are supposed to have normal distributions. The Setup-Time and Delay are assumed to be about 20% more than the mean of setup-time and hold-time, which are simulated by SPICE Monte Carlo simulation. Also, Minimum Inverter Widths and Minimum Pass-transistor Widths are assumed to be 20% less than the minimum possible value for the corresponding probabilistic width. Considering the above assumptions and using the paper [43], in which the area supposed to be the summation of widths of transistors, 500nm is found by scaling the parameters in [43].

According to the ITRS [20], the gate dimension variations are assumed to have a

Table 4.2: Design constraints except area

Constraint	Setup-Time	Hold-Time	Minimum Inverters Widths	Minimum Pass-transistors Widths
Value	2.8×10^{-9}	1.3×10^{-9}	30×10^{-9}	30×10^{-9}

Table 4.3: Experimental results for different threshold voltage variations

Tolerance (%)	Area (nm)	Design Variables (nm)										Yield (%)
		W_{I1}	W_{I2}	W_{I3}	W_{I4}	W_{I5}	W_{I6}	W_{T1}	W_{T2}	W_{T3}	W_{T4}	
12	500	46.8	46.8	46.8	46.8	46.8	46.8	46.8	46.8	46.8	46.8	100
12	480	46.8	46.8	46.8	46.8	46.8	46.8	46.8	46.8	46.8	46.8	98.9
15	500	48.15	48.15	48.15	48.15	48.15	48.15	48.15	48.15	48.15	48.15	99.9
15	480	48.15	48.15	48.15	48.15	48.15	48.15	48.15	48.15	48.15	48.15	56.3

value of 12 % of the physical gate length. The simulations are conducted by using two different numbers as the Area. Fig. 4.6 shows the timing yield maximization results for the first tolerance box width, 12 %, and the area, 500×10^{-9} in two dimensions. For the area equal to 500×10^{-9}, 100 % yield is obtained. It can be seen in Table 4.3 that the sizing of the flip-flop is required to obtain a 100 % yield. Also in Table 4.3, when the the right hand side number in the constraint for Area (first constraint in the main problem) becomes smaller, a smaller yield is attained because feasible region becomes smaller. Also, it can be seen from Table. 4.3 that for higher percentage of tolerances, less yield can be attained. A SPICE Monte Carlo simulation has been done using the nominal values (nominal widths of the gates) of the flip-flop in the problem and confirms that the timing yield, which is found by SPICE Monte Carlo simulation, is close to the timing yield that is found by mathematical simulation of the problem in Matlab. Fig. 4.7 shows the CDF (Cumulative Distribution Function) of the obtained results from SPICE Monte

Carlo simulation. It can be seen in Fig. 4.7 that by using the Setup-Time value from the Table 4.2 (2.8ns), the probability of constraint number 2 (constraint of setup-time) to be valid, is 100% that conveys 100% timing yield.

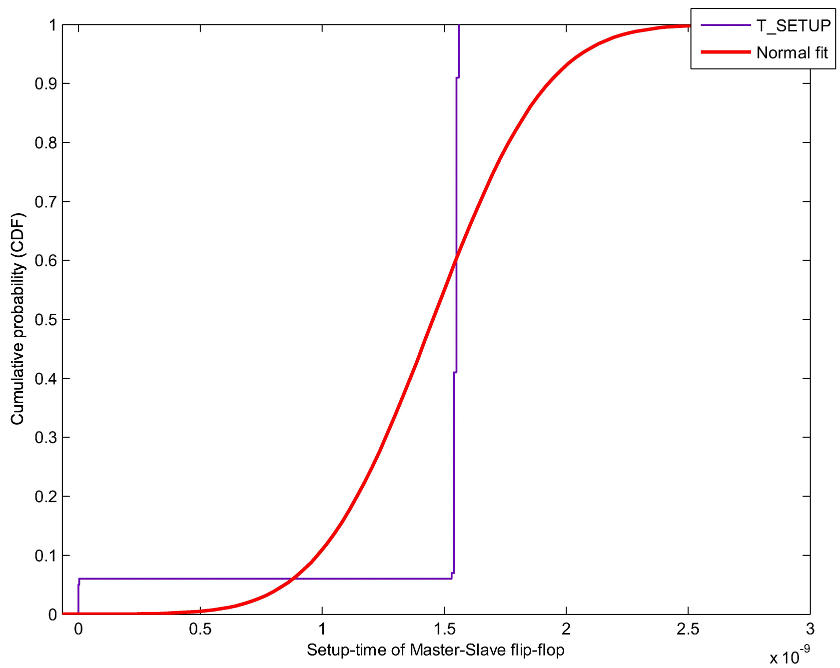


Figure 4.7: CDF of setup-time using the SPICE Monte Carlo simulation, mean = 1.37ns and $\sigma = 458.48ns$

4.5 Conclusion

In this chapter in order to achieve maximum timing yield for setup/hold-time under variation of the widths of transistors, the statistical design of a flip-flop is discussed. This method can also be used for other kinds of flip-flops and digital blocks and with different kind of constraints such as power dissipation and different kind of technologies. It's needed only some changes in the constraints of the statistical problem to be done.

In the next chapter, conclusions for all of the developed methods in this thesis and also suggestion for the future work will be discussed.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

To date, carry look-ahead adders are the fastest, and the Kogge-Stone is viewed as being optimized in terms of delay [15]. However, serial addition techniques surpass other techniques in relation to the power dissipation, but still have the problem of high delay. In this study the proposed Constant-Delay (CD) adder, is a very fast and low power adder that can work faster than the recognized methods for serial additions. As well, it can work with a lower power than that of the famous Kogge-Stone adder. The point of interest is that if the number of operands in the serial addition is slightly more than the word length, the proposed CD adder acts like a constant adder for each single addition during the serial additions. The hardware of the proposed CD adder has a linear relationship with the word length, without sacrificing the area or power, when the delay is optimized. Moreover, the proposed CD technique has a delay that is independent of the length of the words and each bit of the result is calculated separately from the others. Therefore, if many numbers are going to be added in several clocks, some of the blocks can be turned OFF for calculating the most significant bits foregoing the first clocks of serial addition and using the lower width clock, at the first clocks and then turn them ON at the last clocks and use wider clocks at the end. This power saving and delay is not possible in the other methods. After the algorithmic proof for the novel method is introduced, the new method and the Kogge-Stone method are implemented for addition of 64 58-bit numbers. The results confirm that the significant difference between our method and the Kogge-Stone method in serial addition is not only in the delay aspect but also the power issue. In fact, the proposed CD is almost three times faster and consumes half the power of the Kogge-Stone method.

Also in this thesis, a statistical method to flip-flop design is proposed. The method accounts for the manufacturing variability in the transistor widths. In addition, the sizes of flip-flop's gates are chosen such that they satisfy the constraints of performance and area. The newly developed method is flexible and involves small mathematical computations. According to experiments on the proposed flip-flop design, there is no timing yield loss due to gate width variations. The method can be extended to include the impact of other sources of variability such as the oxide thickness fluctuation and line-edge roughness.

5.2 Future work

In this thesis, two robust digital circuit techniques have been probed. In the first case that is talking about serial additions, the future work is the investigation on the parallel and the tree-structure additions. Parallel additions are not energy efficient but they are faster than serial additions. Also, in the second part of the thesis, statistical design of important and most usable blocks in digital circuits can be investigated for the future works.

Bibliography

[1] M. J. Irwin and R. M. Owens, "A case for digit serial VLSI signal processing," *J. VLSI Signal Processing,* no. 1, pp. 321-334, 1990.

[2] S. Cotofana and S. Vassiliadis, "Periodic Symmetric Functions, Serial Addition, and Multiplication with Neural Networks," *IEEE Trans. on Neural Networks* vol. 9, no. 6, November 1998.

[3] V. Beiu, S. Aunet, J. Nyathi, R. R. Rydberg, and W. Ibrahim, "Serial Addition: Locally Connected Architectures," *IEEE Trans. on Circuits and Systems-II: Regular Papers,* vol. 54, no. 11, November 2007.

[4] V. Beiu1, S. Aunet, J. Nyathi1, R. R. Rydberg, and A. Djupdal, "On the Advantages of Serial Architectures for Low-Power Reliable Computations," in *IEEE International Conference on Application-Specific Systems, Architecture Processors,* pp. 276-281, July 2005.

[5] B. Parhami, *Introduction to Parallel Processing, Algorithms and Architectures.* New York: Kluwer Academic Publishers, 2002, pp. 157-161.

[6] J. Sklansky, "Conditional sum addition logic," *IRE Trans. Electron. Comput.*, vol. EC-9, no. 6, pp. 226-231, 1960.

[7] M. J. Irwin and R. M. Owens, "A Comparison of Two Digit Serial VLSI Adders," in *IEEE International Conference on Computer Design: VLSI in Computers and Processors,* pp. 227-229, October 1988.

[8] T. P. Kelliher, R. M. Owens, M. J. Irwin, and T.-T. Hwang, "ELM-A Fast Addition Algorithm Discovered by a Program," in *IEEE Trans. on Computers,* vol. 41, no. 9, September 1992.

[9] G. P. Saggese, A. G. M. Strollo, N. Mazzocca, and D. De Caro, "Shuffled serial adder: an area-latency effective serial adder," in *International Conference on Electronics, Circuits and Systems,* vol. 2, pp. 607-610, September 2002.

[10] Wen-Chang Yeh and Chein-Wei Jen, "Generalized Earliest-First Fast Addition Algorithm," in *IEEE Trans. on Computers,* vol. 52, no. 10, October 2003.

[11] J. D. Nicoud, "B.C.D. serial adder/subtractor," in *Electronics Letters,* vol. 5, pp. 686-687, December 1969.

[12] P. M. Kogge and H. S. Stone, "A parallel algorithm for the efficient solution of a general class of recurrence equations," *IEEE Transactions on Computers*, vol. 22, no. 8, pp. 786-793, Aug. 1973.

[13] S. Knowles, "A family of adders," in *Proc. I5th IEEE Symp. Computer Architecture*, pp. 277-281, June 2001.

[14] R.P. Brent and H.T. Kung, "A regular layout for parallel adders," *IEEE Trans. Computer*, vol. C-31, no. 3, pp. 260-264, Mar. 1982.

[15] S. Das and S. P. Khatri, "A Novel Hybrid Parallel-Prefix Adder Architecture with Efficient Timing-Area Characteristic," in *IEEE Trans. VLSI Systems*, vol. 16, no. 3, March 2008.

[16] H. Zhu, C. K. Cheng, and R. Graham, "On the construction of zerodeficiency parallel prefix circuits with minimum depth," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 11, no. 2, pp. 387-409, 2006.

[17] G. Dimitrakopoulos and D. Nikolos, "High-speed parallel-prefix VLSI ling adders," *IEEE Trans. Comput.*, vol. 54, no. 2, pp. 225-231, Feb. 2005.

[18] V. Dave, E. Oruklu, and J. Saniie, "Performance evaluation of flagged prefix adders for constant addition," in *Proc. IEEE Int. Conf. Electro/ Inf. Technol.*, 2006, pp. 415-420.

[19] Vojin G. Oklobdzija, Bart R. Zeydel, Hoang Q. Dao, Sanu Mathew, and Ram Krishnamurthy, "Comparison of High-Performance VLSI Adders in the Energy-Delay Space," in *IEEE Trans. VLSI Systems,* vol. 13, no. 6, June 2005.

[20] International Technology Roadmap for Semiconductors, http://www.itrs.net/, 2006.

[21] Vishvanathan, C. P. Ravikumar, and Vinod, Menezes, "Design Technology Challenges in the Sub-100 Nanometer Era," in *The periodical of the VLSI society of India - VLSI Vision,* vol. 1, no. 1, 2005.

[22] M. Hansson, A. Alvandpour, S. K. Hsu, and R. K. Krishnamurthy, "A process variation tolerant technique for sub-70nm latches and llip-flops", In *Proc. of NORCHIP,* pp. 149-152, 2005.

[23] S. Hatami, H. Abrishami, M. Pedram, "Statistical Timing Analysis of Flip-flops Considering Codependent Setup and Hold Times," In *Proc. Of GLSVLSI,* pp. 101-106, 2008.

[24] J. P. Fishburn, "Clock skew optimization," *IEEE Transactions on Computers,* 1990, vol. 39, pp. 945-951.

[25] C. Lin, H. Zhou, "Clock skew scheduling with delay padding for prescribed skew domains," In *Proc. of ASP-DAC,* pp. 541-546, 2007.

[26] G. Venkataraman, C. N. Sze, and J. Hu, "Skew scheduling and clock routing for improved tolerance to process variation," In *Proc. of ASP-DAC,* pp. 594-599, 2005.

[27] A. P. Hurst and R. K. Brayton, "Latch based design under process variation," In *Proc. of IWLS,* 2006.

[28] B. Voss and M. Glesner, "A low Power Sinusoidal Clock," *IEEE Internaional Symposium on Circuits and Systems,* vol. 4, pp. 108-111, 2001.

[29] Hassan Mostafa, Mohab Anis, and Mohamed Elmasry, "Comparative Analysis of Timing Yield Improvement under Process Variations of Flip-Flops Circuits," *IEEE Computer Society Annual Symposium on VLSI (ISVLSI 2009),* pp.133-138, 2009.

[30] V. Stojanovic, and V. G. Oklobdzija, "Comprehensive Analysis of Master-Slave Latches and Flip-Flops for High Performance and Low Power Systems," *IEEE Journal of Solid-State Circuits,* vol.34, no. 4, pp. 536-548, 1999.

[31] H. R. Srinivas and Keshab K. Parhi, "A Fast VLSI Adder Architecture," in *IEEE Journal of Solid-State Circuits,* vol 21. no. 5 . May 1992.

[32] H. R. Srinivas, Bapiraju Vinnakota, and Keshab K. Parhi, "A C-Testable Carry-Free Divider," in *IEEE Trans. on Very Large Scale Integration (VLSI) Systems,* vol. 2, no. 4. December 1994.

[33] S. Ghosh, P. Ndai, and K. Roy, "A Novel Low Overhead Fault Tolerant Kogge-Stone Adder Using Adaptive Clocking," *Design, Automation and Test in Europe*, pp. 366-371, March 2008.

[34] Xiaoping Huang, Wen-Jung Liu, and Belle W. Y. Wei, "A High-Performance CMOS Redundant Binary Multiplication-and-Accumulation (MAC) Unit," in *IEEE Transactions on Circuits and Systems-I,* vol. 41, no. 1, January 1994.

[35] Masakazu Yamashina, Junichi Goto, Fuyuki Okamoto, Kouichi Ando, Hachiro Yamada, Tadahiko Horiuchi, Kimiko Nakamura, and Tadayoshi Enomoto, "A 200-MHZ 16-bit super high-speed signal processor (SSSP) LSI, in *IEEE J. Solid-state Circuits,* vol. 24, Dec. 1989.

[36] Junichi Goto, Kouichi Ando, Toshiaki Inoue, Masaki Ishida, Masakazu Yamashina, Hachiro Yamada, and Tadayoshi Enomoto, "A 250-MHZ 16b

I-million transistor BiCMOS super-high-speed video signal processor, in *ISSCC Dig. of Tech. Papers,* pp. 254-255, Feb. 1991.

[37] Predictive Technology Models for 45nm, "http://www.eas.asu.edu/ ptm."

[38] J. Rabaey, A. Chandrakasan and B. Nikolic, *Digital Integrated Circuits: A Design Perspective,* 2/e, Prentice Hall, 2003.

[39] A. Seifi, K. Ponnambalam and J. Vlach, "Maximization of manufacturing yield of systems with arbitrary distributions of component values", *Annuals of Operational Research (99) 2000,* pp 373-383.

[40] K. Ponnambalam, A. Seifi, and J. Vlach, "Probabilistic design of systems with general distributions of parameters", *Intl' J. of Circuit Theory and Appl.* 29(6) 2001, pp 527-536.

[41] S. Director, P. Feldmann,"Optimization of Parametric Yield: A tutorial," *IEEE Custom Integrated Circuits Conference,* 1992, pp. 3.1.1-3.1.8.

[42] P. Kumaraswamy, "A generalized probability density function for double-bounded random processes," *Journal of Hydrology(46),* 1980, pp. 79-88.

[43] V. Stojanovic, and V. G. Oklobdjija, "Comparative Analysis of Master-Slave Latches and Flip-Flops for High Performance and Low Power Systems," *IEEE Journal of Solid-State Circuits,* vol. 34, no. 5, pp. 712-716, 1999.