AUTOMATED SYNTHESIS OF APPROXIMATE CIRCUITS FOR ERROR RESILIENT APPLICATIONS

A Project Report

submitted by

CHAITRA YARLAGADDA

in partial fulfilment of the requirements
for the award of the degree of

BACHELOR OF TECHNOLOGY
and
MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

MAY 2014

DEDICATION

To my family

and

IIT Madras

THESIS CERTIFICATE

This is to certify that the thesis titled **Automated Synthesis of Approximate Circuits for Error Resilient Applications**, submitted by **Chaitra Yarlagadda**, to the Indian Institute of Technology, Madras, for the award of the degree of **Bachelor of Technology** and **Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof.Shankar Balachandran
Research Guide
Assistant Professor
Dept. of Computer Science
IIT-Madras, 600 036

Place: Chennai

Date: 16th May 2014

ACKNOWLEDGEMENTS

I am very grateful to my guide Prof. Shankar Balachandran for giving me the opportunity to work under him and lending constant support at every stage of this project. I truly appreciate and value his guidance and encouragement throughout the project. I would like to thank Prof. Nitin Chandrachoodan, Prof.Sridharan and Prof. Kamakoti for providing the necessary background needed for this research through their courses. I am indebted to various faculty members of IIT Madras from whom I have benefited as a student. I would like to thank my parents, sister and friends who have been very understanding and supporting me not only during this project, but also throughout my five years stay here and I dedicate this thesis to them.

ABSTRACT

KEYWORDS: Low power design; Approximate Computing; Functional Approximation.

With the advent of new generation portable devices, with computationally heavy and complex processes consuming a great deal of the power and area, low power design has become an imperative requirement. Most multimedia applications today are error resilient; they give outputs of reasonable quality irrespective of erroneous computations. In image, video and audio applications, the insensitiveness of human perception to small amount of errors increases the scope for approximating circuits as we do not need to produce precise outputs. Previous research in this context exploits resiliency primarily through voltage overscaling, utilizing algorithmic and architectural techniques to mitigate errors. There have been efforts to functionally approximate circuits, but these were focused on manual design techniques. We demonstrate a methodology based on Venkataramani *et al.* (2012) to automate the process of generating approximate circuits, given the RTL specification of the exact circuit and a quality constraint that defines the amount of error that can be tolerated. We have implemented this technique using two logic synthesis tools âĂŞ SIS and Synopsys Design Compiler. We use this automation technique on a set of arithmetic and standard MCNC benchmark circuits to generate their corresponding approximate versions. We demonstrate the scalability of the technique and significant improvements in area and power. The technique yields approximate circuits that lead to area savings in the range of 1.1X-1.3X for strict error bounds and 2.4X for relaxed error bounds and power savings in the range of 1.1X-1.2X for tight error bounds and 2.5X for relaxed error bounds with respect to their exact circuit counterparts. Furthermore, we demonstrate the utility of the generated approximate circuits in image processing applications with specific quality constraints.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

ADC	Approximate Don't Care
BLIF	Berkeley Logic Interchange Format
CLA	Carry Lookahead Adder
CMOS	Complementary Metal Oxide Semiconductor
DC	Design Compiler
EXDC	External Don't Care
FFT	Fast Fourier Transform
MCNC	Microelectronics Center of North Carolina
ODC	Observability Don't Care
RCA	Ripple Carry Adder
SALSA	Systematic methodology for Automatic Logic Synthesis of Approximate circuits
SIS	Sequential Interative Synthesis
VLSI	Very Large Scale Integration
WTM	Wallace Tree Multiplier

CHAPTER 1

INTRODUCTION

Semiconductor technology has been undergoing tremendous changes ever since the first integrated circuit (IC) was introduced in the 1960s. IC design so far has been focused on high performance computing and small chip area. However, as portable device market increases exponentially, low power design is becoming an equally important design consideration. Figure 1.1 shows the exponential growth in the number of global cellular phone subscribers, and Figure 1.2 shows the predicted sale of media tablet products. Power consumption is directly related to the runtime of the battery operated devices as well as the weight and volume of the devices. International technology roadmap for semiconductor (ITRS) reported that the battery life has declined as more and more new features are added into the devices while the development of battery technology is much slower than the increase in the functional requirements. In this context, low power design techniques have become the need of the hour.

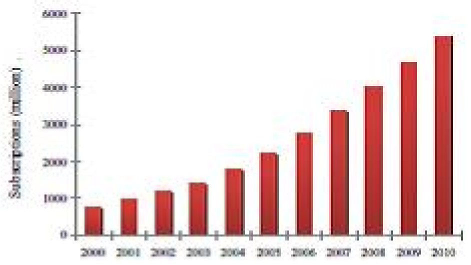
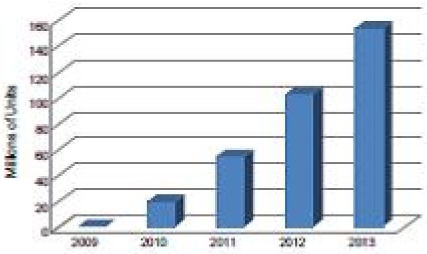


Figure 1.1: Global growth in the number of mobile cellular phone subscribers



{

Figure 1.2: Global media tablet sale forecast

VLSI designers have always tried to improve the performance of the systems. Although todays systems are much faster and far more versatile than their predecessors, they also consume a lot of more of power, so much power, in fact, that their power densities and concomitant heat generation are rapidly approaching levels comparable to nuclear reactors. These high power densities impair chip reliability and life expectancy, increase cooling and packaging costs, and will pose the problem of battery life in case of portable devices. Although one could equip the portable devices with faster processors and larger memories, this would diminish their battery life even further. Without cost effective solutions to the power problem, improvements in micro-processor technology will eventually reach a standstill **?**. Thus low power design has become a significant design metric in this era of portable devices. It has emerged as a principal theme in todays electronics industry. The need for low power design has caused a major paradigm shift where power dissipation has become as important a consideration as performance and area Pedram (1997).

In the past, power was given secondary priority by chip designers while area, performance, cost effectiveness and reliability have been considered the main drivers of design. However the scenario has begun to change and power has gained importance in the design consideration as portable devices have taken over the market. With the advent of personal computing devices and wireless communication systems, which has been one of the most successful disruptive technologies of present time, market has opened to new innovative measures to reduce power consumption, as these devices demand for high-speed computation and complex functionality with low power consumption. These high performance systems consume high power as they function at high frequency. System integration and bandwidth further add to it. Thus power has become the real price of performance. In the absence of innovations in design techniques to lower power consumption, future portable devices will suffer from either short battery life or bulky battery pack. Consumer demand played a great role in this paradigm shift towards an increased focus on reducing power. Users expect new products to be more powerful, yet run twice as long as before on battery power.

Additionally, reducing the power consumption in high performance systems will provide a great financial advantage. Hence, the producers of high-end products are pressed to undertake measures to lower power consumption. The power consumed by the core should be dissipated through the packaging and thus a circuit with high power consumption would imply high cost to remove heat at packaging level. The cost associated with packaging and cooling can be decreased considerably by employing low power techniques.

High power consumption leads to high heat dissipation. Higher temperature not only affects circuit performance directly by slowing down the transistors on CMOS chips but also decreases their reliability Chu and Wong (1998). Every 10ÂřC rise in operating temperature roughly doubles a circuitâĂŹs failure rate Small (1994). High power consumption limits the number of transistors that can be integrated on a chip as thermal management comes up as an important issue in shrinking designs. In this context, peak power (maximum possible power dissipation) is a critical design factor as it determines the thermal and electrical limits of designs, impacts the system cost, size and weight, dictates specific battery type, component and system packaging and heat sinks, and aggravates the resistive and inductive voltage drop problems. It is therefore essential to have the peak power under control Pedram (1997).

These crucial factors have led to an increased emphasis on low power VLSI design techniques. However, the motivation varies from application to application. In the case of high performance portable computers and laptops, the innovations are aimed at reducing the power dissipation of the electronics of the system. While in the case of micro-powered, portable applications, such as mobile phones and personal digital assistants, innovations are aimed at improved battery lifetime, reduced weight and low packing cost. Low power levels will enable the use of inexpensive plastic packages. Power minimization in high performance, non-battery operated systems is aimed at reducing cost in terms of packaging, cooling and energy bill and ensuring long-term reliability. Thus, based on the application the designer can decide how much he is willing to sacrifice in cost or performance to obtain lower power dissipation and can choose a low power technique accordingly.

The low power techniques can broadly be divided into two categories: the ones that introduce errors and the ones that do not. As more and more multimedia applications are integrated into portable devices, especially mobile phone, a significant research emphasis has been laid on minimizing power in digital signal processing and image processing applications. Numerous techniques for low energy DSP and IP systems have been sought at various levels of design such as algorithm, architecture and circuit. There have been many approaches over the years aiming to reduce the power consumption, e.g., scaling voltage supply Gonzalez *et al.* (1997),**?**, reducing unneeded circuit activity through clock-gating Téllez *et al.* (1995),Wu *et al.* (2000), scaling down transistor sizes or minimizing chip area through synthesis optimizations Micheli (1994), and developing novel power-efficient transistors Hisamoto *et al.* (2000), Kao *et al.* (1997). All these approaches attempt to reduce power consumption without any impact on the correctness of the output quality and the functioning of the chip.

In contrast, techniques based on error resilience of systems explore energy saving techniques by trading off the accuracy of computations. The output quality of multimedia applications is often determined by human perceptual system, which masks small errors Wang and Bovik (2006). This brings inherent error tolerance for such applications. In this thesis, we take advantage of this inherent imprecision tolerance of systems to approximate circuits, and arrive at low power circuits at the cost precision. We further demonstrate the usage of these circuits in multimedia and show that with a little, hardly noticeable imprecision, we can minimize the power consumption by a great deal.

1.1 Contributions of this thesis

This thesis proposes approximate logic circuits as a new logic design paradigm for designing power efficient, reliable computing systems. We explore the technique of approximate computing, making using of the inherent error resilience that exists in the systems to reduce area and power. We demonstrate a methodology based on Venkataramani *et al.* (2013) to synthesize functionally approximate circuits. We assess the technique by evaluating the power-error and area-error metrics on some standard arithmetic and MCNC logic synthesis benchmarks. We demonstrate the utility of the synthesized approximate circuits in certain Image Processing Applications.

1.2 Organization of the thesis

This thesis is organized into five chapters. Chapter 1 introduces the paradigm shift to low power design and summarizes the contributions of this thesis.

Chapter 2 explains the necessary background materials required to gain a clear understanding of the thesis.

Chapter 3 reviews related work and recent progress in approximate computing and introduces SALSA.

Chapter 4 presents the methodology used to arrive at an approximate circuit.

Chapter 5 describes how to set up experiments to verify the proposed method and presents the results.

Chapter 6 gives conclusions (summarizes the major accomplishments of this thesis) and suggests future research.

CHAPTER 2

Background

2.1 Introduction

This chapter provides all the necessary information and disjoint background materials directly related to this work to render this document self-contained and to appreciate the problems discussed. Since our work is focused on low power design methodology, we start with some basics about the sources of power dissipation. Then, we explain the concept of approximate computing and talk about the systems where it can be implemented. We then describe the various logic minimization and synthesis tools used for evaluating the proposed ideas and the benchmarks used to assess the technique.

2.2 Sources of Power Dissipation

Power dissipation in CMOS circuits can be attributed to three sources: the leakage current, short circuit current and the charging-discharging of capacitive loads during logic changes. The total power consumption is thus given by the equation below.

$$P_{total} = P_{leakage} + P_{short-circuit} + P_{switching} \tag{2.1a}$$

The leakage power consumption is classified as static power consumption and the short-circuit power consumption and switching power consumption are classified as dynamic power consumption.

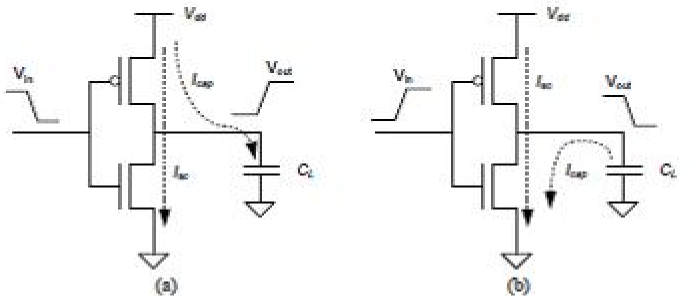


Figure 2.1: Illustration of sources of dynamic power consumption

The leakage power consumption is due to leakage current ($I_{leakage}$) that includes subthreshold and gate leakage currents. The leakage occurs when a transistor is turned off and another active transistor charges up or down the drain with respect to the first transistorâĂŹs bulk potential Pedram (1997). As technology feature size shrinks, supply voltage has been scaling down to keep power consumption under control. To maintain performance, threshold voltage has been scaled accordingly. In addition to the scaling of channel lengths, gate-oxide has been scaling down as thin as possible to increase channel conductivity and performance. This reduction in threshold voltage and gate-oxide scaling increases the leakage current. In Chandrakasan *et al.* (2000), the subthreshold leakage current is expressed as

$$I_{rub} = I_0 e^{(}\frac{V_{gx} - V_t}{nV_{TH}})(1 - e^{\frac{-V_{dx}}{V_{TH}}}) \tag{2.2a}$$

This leakage current is typically 1 picoA for a 1 micro-meter minimum feature size. The sub-threshold current will remain 102-105 times smaller than the âĂIJon currentâĂİ even at submicron device sizes Pedram (1997).

Dynamic power consumption arises from circuit activity such as changes in inputs or values in a register. It has two sources, short circuit current and the charging-discharging of capacitive loads.

When we talk about ideal CMOS circuits, we assume that pull-up and pull-down networks never conduct simultaneously. However, in reality, when the output of a gate changes and pull-up and pull-down networks are both ON during the change for a short period, the current (I_sc) flows directly from Vdd to the ground terminal and has no contribution towards charging of the output capacitance. Power consumption due to this current is known as short-circuit power consumption. This short-circuit current is only the secondary source of dynamic power and by choosing gate sizes such that the input and output rise/fall times are about equal, the short-circuit power consumption can be made less than 15

The short-circuit and leakage currents in CMOS circuits can be minimized by employing proper circuit and device design techniques. It is the charging and discharging of capacitive loads that leads to major power dissipation in a circuit. Thus, switching power is the primary source of dynamic power consumption and arises from the charging and discharging of capacitive loads through normal circuit operations during logic changes. Consider an inverter as shown in figure below. During the falling transition at the input node, the current (I_cap) flows through the PMOS from the power supply to the output node and the load capacitance (CL) is charged. This stored charge is dissipated through the NMOS during the raising transition at the input node. The average switching power consumption is given by

$$P_{switching} = \alpha_{0->1} f_{clk} C_L V_{dd}^2 \tag{2.3a}$$

where $\alpha_{0->1}$ is the switching activity factor, f_{clk} is the average frequency of zero-to-one transitions, C_L is the load capacitance of the circuit and V_{dd} is the supply voltage.

2.3 Approximate Computing for Low Power Design

It is now clear that the fundamental component of power dissipation in circuits is the power lost at capacitive loads due to switching activity. Low power VLSI design can be achieved at various levels of design abstraction from algorithmic and system levels down to layout and circuit levels. By relaxing the computational accuracy at various levels of abstraction, a designer can make the systems more power efficient as approximating circuits or logic design will reduce the switching activity and thus the dynamic power. In fact, by arriving at clever and careful approximations, one can reduce the number of logic gates and thus the transistors in a circuit with a little compromise in the performance. This will reduce the leakage current (as there would be lesser number of transistors) and also the short circuit current if designed carefully.

The new generation of portable devices run heavy multimedia applications which use image and video processing, pattern recognition and data mining. These applications are termed $heavy$ as they not demand complex signal processing of data to achieve high quality, but also consume a great amount of power. Here lies the major design challenge as the designer should arrive at power efficient circuits that produce high quality outputs. However, these systems offer an advantage to the designer in terms of their inherent error-resilience. Error resilience can be broadly defined as the characteristic of an application to produce acceptable outputs despite its constituent computations being performed imperfectly (with errors) Venkataramani *et al.* (2012). This intrinsic error tolerance in systems stems from a number of sources. Some of the factors that can be attributed for error tolerance in systems are:

- insensitiveness of human perception to small amount of errors in output,
- redundancies in large-input data-sets,
- non-existence of unique golden result in some cases,
- aggregating nature of algorithms which lead to averaging out the error,
- usage of statistical and probabilistic computations in several cases.

There are several emerging applications that show this kind of error resilient behaviour. For example, if the degree of brightness of one or two pixels in a 500 by 500 image is not very precise, human brain will not be able to sense this error. This kind of perceptual resilience to errors is exhibited by applications that involve a human interface. Multimedia applications in video, audio and graphics processing domains fall under this category of systems, they can tolerate errors that are insensitive to human perception. This property of multimedia systems has been detailed in Breuer (2005), where the author states that the numerical results produced by a digital system in such domains needs to be just $goodenough$, and not necessarily exactly right all the time. Other applications that manifest error resilience have been detailed in Chen *et al.* (2008). These include machine learning and RMS (recognition, mining and synthesis). These systems are error tolerant because of the redundancies in large input data-sets, non-existence of golden result and aggregating nature of algorithms which lead to averaging out the error Chakradhar and Raghunathan (2010). RMS finds application in several market segments such as gaming, graphics, media-mining and financial analytics. Thus we have a whole new set of emerging applications that show the property of error resilience. Manufacturers of high performance devices disable some blocks of logic that are faulty. This has led to increased difficulty in maintaining good yield with decreasing feature size. This in turn increases their price and slows down the transition to a new technology. Breuer (2005) states that as long as a chip with certain defects produces infrequent erroneous responses of small magnitude with little impact on system performance, it need not be discarded. Due to the error resilient behaviour of systems, devices are no longer expected to behave in a deterministic and reliable manner. In fact, the probabilistic and unreliable behavior of devices is deemed inevitable by the international technology road-map for semiconductors (itrs) which forecasts Allan *et al.* (2002) that relaxing the requirement of 100 percent correctness for devices and interconnects may dramatically reduce costs of manufacturing, verification, and test. Such a paradigm shift is likely forced in any case by technology scaling, which leads to more transient and permanent failures of signals, logic values, devices, and interconnects.

Designers can exploit the error resilient property of the emerging systems, by relaxing the computational accuracy. Approximation can be introduced at various levels of design by relaxing equivalence across layers of design abstraction. Trading precision to arrive at low power circuits through careful designs is what constitutes Approximate Computing. Approximate computing leverages the intrinsic resilience of applications to inexactness in their computations, to achieve a desirable trade-off between performance and acceptable quality of results Venkataramani *et al.* (2013). The span of approximate computing is vast. In simple words, it can be termed as the use of imprecise hardware designs and approximate algorithms to arrive at energy efficient systems. There are two major schools of thoughts when it comes to the design of approximate circuits. Techniques where voltage scaling is used to induce timing errors in circuits are classified as over-scaling based approximation techniques. Techniques aimed at reducing the implementation complexity of circuits by approximating logic functions are termed as functional approximation techniques Venkataramani *et al.* (2012). The recent progress pertaining to these two techniques has been described in Chapter 3.

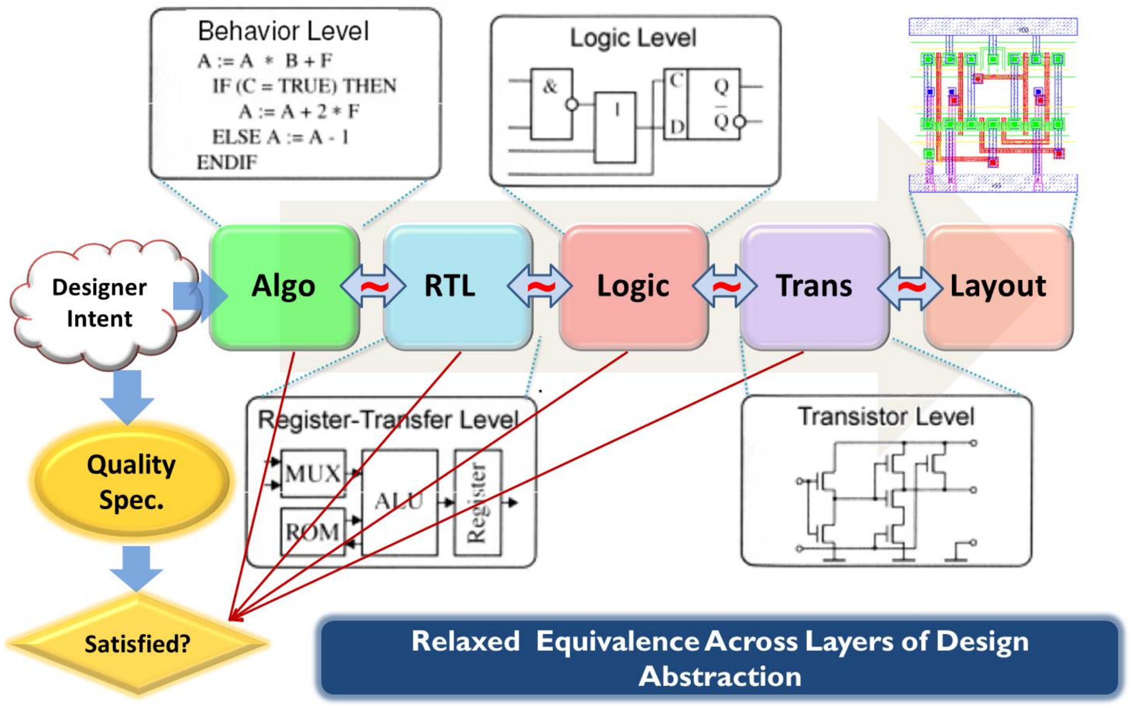


Figure 2.2: Approximate Design

2.4 Description of Synthesis Tools and Benchmarks used

2.4.1 SIS

SIS is a logic synthesis tool for combinational and sequential circuits. It takes as input a representation of a set of logic functions. The two most common forms of input are a finite state machine and a netlist of gates and latches. In our experiments, we restricted the input to a netlist representation. The input format, an intermediate level description between a hardware description language like Verilog and a netlist of standard gates, is called BLIF (Berkeley Logic Interchange Format). The BLIF file is a representation of a multi-level logic network which consists of interconnected nodes and latches. Each node has a boolean function, whose variables are the fanins of the node, associated with it. This function could be arbitrarily complex. The goal of SIS is to optimize the network with respect to performance or area.

2.4.2 MCNC Circuits

MCNC (Microelectronics Centre of North Carolina) benchmark suite was developed for the MCNC International Workshop on Logic Synthesis 1991 (IWLS âĂŹ91). The suite includes logic synthesis and optimization benchmarks collected from industry and academics. It also includes some benchmarks developed in other workshops and conferences. The designs in the MCNC suite range from simple circuits to very advanced circuits. Yang (1991) provides adetailed information on the MCNC benchmark suite. The MCNC suite contains around 200 combinational circuits. The designs are available in many different circuit formats such as pla, blif, etc. However, some circuits are very large and hence cannot be inter-converted from one format to another using the existing academic tools. Hence we use the appropriate subset of MCNC benchmarks as applicable according to the problem.

CHAPTER 3

Related Work and Recent Progress

The previous chapter explained the concept of approximate computing. This chapter reviews the recent progress in approximate computing and related areas and describes design solutions aimed at redesigning hardware to exploit the benefits of approximate computing. We give a brief overview about various over-scaling based approximation and functional approximation techniques. We then describe Venkataramani *et al.* (2012), which is the primary focus of this thesis.

3.1 Over-scaling based approximation techniques

From our discussion in Chapter 2.1, it is evident that supply voltage (V_{dd}) is directly related to the three components of power consumption. Scaling down supply voltage will not only lead to a quadratic reduction in switching power, but also leads to a reduction in short circuit and leakage power. Therefore, scaling supply voltage might seem to be an efficient technique for low power design. However, scaling voltage leads to an increase in propagation delay. The relationship between circuit delay (τ_d) and supply voltage (V_{dd}) is given by equation below Gonzalez *et al.* (1997) where K is a process dependent constant and alpha varies between 1 and 2.

$$\tau_d = \frac{C_L V_{dd}}{K(V_{dd} - V_t)^{\alpha}} \tag{3.1a}$$

The increase in delay caused by scaling down voltage might induce error in the output. Hence, this approximation technique can be applied only to error tolerant systems. The minimum possible supply voltage required to ensure the correct functioning of the circuit is termed as the critical supply voltage. A margin on critical supply voltage is set considering the environment and process related variations that might impact the circuit performance. Several process variations such doping and gate-length and unexpected voltage drops, temperature variations, etc. may have an impact on the circuit performance. However, most of these variations are either data dependent, meaning, they lead to their worst case damage on circuit only under certain instruction sequences or the variations impact only a certain region of the die. The margin on critical voltage is set high to ensure exact output even during the worst-case combination of all the possible variations. However, the occurrence of such worst-case combination is very rare. Keeping in mind the rare occurrence of such an event, the supply voltage can be scaled down thereby giving way to some loss in precision or performance.

Scaling supply voltage leads to rapid quality loss due to timing errors. To correct such errors and still maintain low voltage, suitable corrective mechanisms can be introduced into the system such that it can tolerate the timing errors. Another way is to dynamically modulate the supply voltage based on the performance and quality demand. Yet another way is to control the supply voltage based on error rate through a feedback loop. This approach, known widely as the RAZOR approach has been presented in Ernst *et al.* (2003).

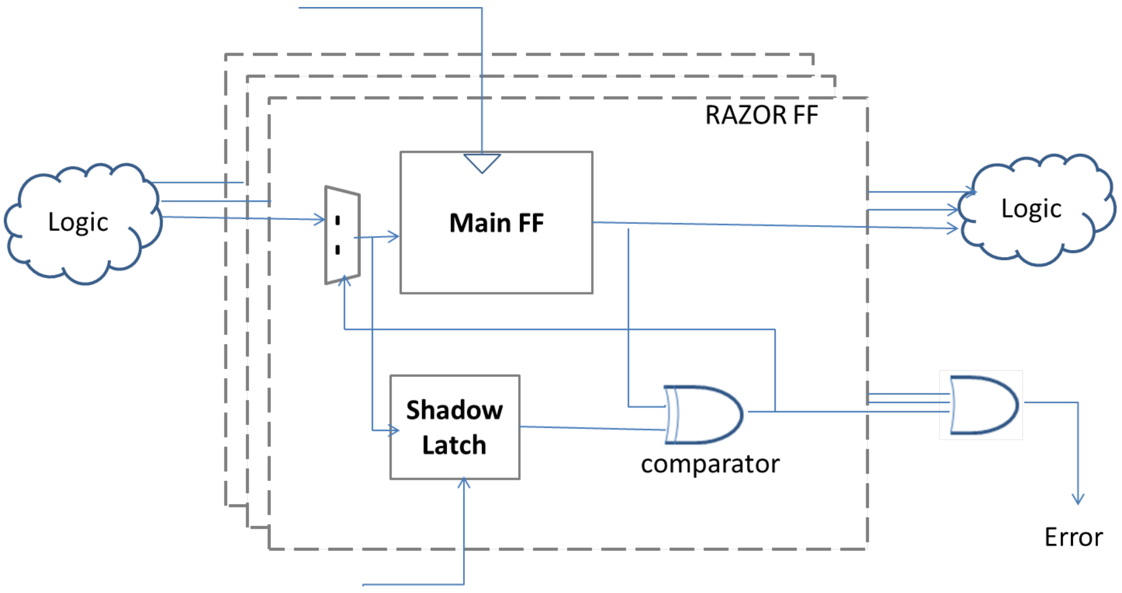


Figure 3.1: RAZOR flip-flop architecture

The approach includes in-situ timing error detection using a shadow latch, which captures a signal at the negative edge while a normal flip-flop does at the positive edge. (fig.3.1) shows the RAZOR flip-flop architecture. In the presence of a timing failure due to increased delay under aggressive voltage scaling, the shadow latch still can capture correct value. Therefore, by comparing the captured signal at an original flip-flop to that at the shadow latch, timing failures can be detected easily. Whenever an error is detected, correct computation is achieved using replay mechanisms in pipelined micro-architectures. This approach is based on the assumption that the error occurrence is not very frequent and highly data dependent since it is directly related to performance degradation. By monitoring error rate, the supply voltage is controlled dynamically to achieve optimal energy savings with minimum performance penalty. Since this error control technique is focused on pipelined micro-architectures and provides error-free results, this technique is suitable for general purpose computing architectures.

3.2 Functional Approximation Techniques

Approximating logic functions, would lead to lesser number of transistors or gates, shorter critical path, reduced switching activity and lower leakage power. Functional approximation techniques involve approximations of Boolean functions of the underlying cores in computing circuits for less complex design implementations. Previous attempts to design such functionally approximate circuits focused on manual step by step re-design of specific classes of circuits and arithmetic functions Gupta *et al.* (2013). Approximation techniques for a wide range of adders Gupta *et al.* (2011),Gupta *et al.* (2013) and Shin and Gupta (2008) and multipliers Kulkarni *et al.* (2011) have been designed in the past by taking advantage of the structural properties of this building blocks and the significance of their output bits.

Numerous such techniques have been proposed in recent times. These existing circuit approximation techniques are aimed at re-designing specific, simple standard building-block circuit components such as adders and multipliers Gupta *et al.* (2011). The approximate circuits in these cases were generated by either low-level modifications or through application-specific modifications. With the recent advances in machine learning and multimedia applications, circuits have become increasingly complex and power consuming. The existing conventional methods do not offer many benefits in such cases and should be replaced by automated approximation techniques that can be applied to multi-level circuits. With this increasing complexity of circuits, the logic approximations that can be performed become non-intuitive. Also, most of the existing functional approximation techniques are specific to adders, a few of them are specific to multiplier. Hence, an automated synthesis procedure that can be applied to a wide range of circuits is needed to approximate more complex circuits.

The first automation effort on these lines focused on identifying minterm complements that produce an approximate circuit version that has the smallest number of literals for a given threshold of error rate Shin and Gupta (2010). This technique is applicable only to two level circuits. In Shin and Gupta (2011), a pre-specified bound is set and circuits are simplified by propagating stuck-at-faults across the nodes. A modified automatic test pattern generation (ATPG) algorithm is accordingly used to estimate the errors and this process iteration is repeated until the pre-specified bounds are violated. In Lingamneni *et al.* (2011), probabilistic pruning is used to arrive at inexact circuits. A new logic synthesis based CAD framework that incorporates this technique is discussed in Lingamneni *et al.* (2011).

The techniques discussed above require the design of a custom CAD tool to perform the required logic approximations. Another common attribute of these techniques is that they only simplify circuits by propagating redundancies and do not alter the structure of the circuit. Also, all these techniques require iterative simulations to ensure that the synthesized inexact circuit adheres to the error quality bounds.

A more recent technique to synthesize approximate circuits is discussed in Venkataramani *et al.* (2012), henceforth referred to as SALSA in this thesis.

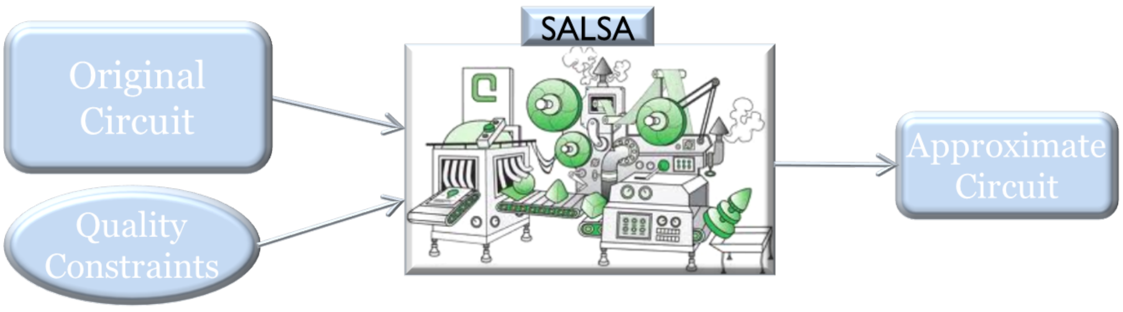


Figure 3.2: SALSA

In SALSA, the problem of synthesizing approximate circuits through approximate logic synthesis is reformulated using circuit transformations and is mapped to traditional logic synthesis problem. This enables the use of existing logic synthesis tools to solve the problem. Thus a wide range of approximations can be applied making use of the well formulated and tested techniques that are often used in logic synthesis tools. This technique not only prunes the gates to approximate the logic, but also transforms the functionality of circuit nodes, if needed. Most importantly, SALSA avoids the need for iterative simulations and checking for error bounds as approximation of logic begins only after providing an inherent guarantee that the synthesized inexact circuit does not violate the error bounds. This makes the method more adaptable and generic, and enables SALSA to be applied to a wide range of circuits and applications. Hence, in this thesis, we have choose a methodology based on SALSA to synthesize approximate circuits and validate the results by showing the use of these circuits in some applications.

CHAPTER 4

Methodology of Automated Synthesis of Approximate Circuits

Let us first put across the problem statement for the synthesis of functional approximation of circuits. Given the RTL description of the logic circuit and the error bounds that can be tolerated, the automated synthesis technique should perform logic simplification and produce a circuit with approximated logic that is well within the error bounds. In this chapter, we describe the methodology used to automate the synthesis of functionally approximate circuits.

4.1 Quality Function

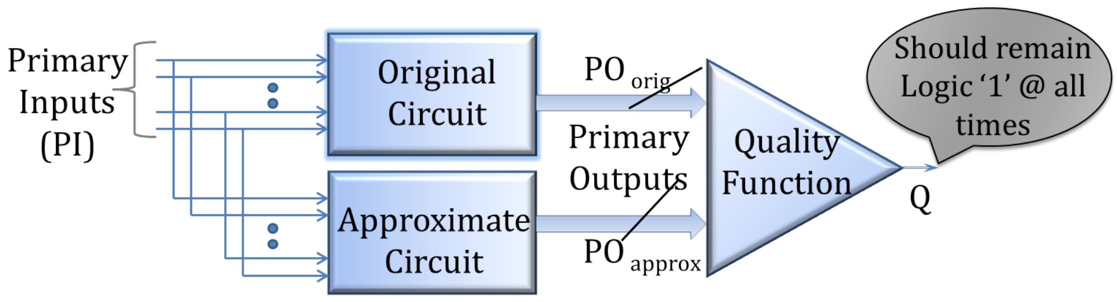


Figure 4.1: Quality Constraint Block

Let us first discuss about the Quality Constraint Block that is used to set the error bounds. As shown in figure above, the Quality function takes inputs from two blocks, the original circuit block and the approximate circuit block. According to the problem statement that has been defined in the beginning of this chapter, the original circuit is available as an input. This original circuit block contains the structural description of the exact circuit on which the approximation technique is to be applied. We also have a pre-defined error bound available and this is encoded in the Quality function. We should now devise a technique to synthesize the approximate circuit block, which is to contain the structural description of the inexact or approximate circuit, ensuring that the error between the original and approximate block outputs is always within the bound that has been pre-set. This error bound is defined with the help of a Quality function as shown in equation below.

$$Q = (|PO_{orig} - PO_{approx}| \leq K)?1 : 0 \tag{4.1a}$$

The Quality function block ensures that for each set of primary inputs, the difference in the values as given by the primary outputs of the exact and approximate circuitsâĂŹ lies within the error bound, K. The output of the Quality function block, Q should be 1 for the approximate circuit outputs to lie within the error bound. Throughout the process of our synthesis, we preserve this invariant.

With the help of this constraint, we identify Observability Dont Cares (ODCs) and use traditional logic synthesis approach to simplify the logic. The details have been described in the following section.

4.2 Approximation Don't Cares

With respect to the Quality Constraint Block, the primary outputs of the original and approximate circuits are internal nodes. According to our problem statement, we are allowed to functionally modify the approximate circuit as long as the error bound is not violated. In other words, we can make modify our approximate circuit such that the changes do not affect the value of Q which should be maintained at 1 to ensure that the error constraint is met.

We now explain how the approximate logic synthesis problem is transformed into a traditional logic synthesis problem.In Micheli (1994), Observability Dont Cares (ODCs) of a node in a logic circuit are defined as the set of input values for which the primary outputs of the circuit remain insensitive to the nodeâĂŹs output. In simple words, the input combinations that do not affect the primary outputs in the circuit are classified as ODCs and can be used to simplify the node. PO$_{approx}$ is an internal signal in the

Quality function block. We can find the ODCs in this block, which are nothing but the primary inputs for which Q is insensitive to one of PO$_{approx}$ bits. We use this information to functionally approximate the circuit and hence these ODCs are termed as $Approximate\ Dont\ Cares$ (ADCs) of the circuit.

In traditional logic synthesis tools, we use External Dont Cares (EXDCs) to simplify circuits. We use this principle in our approximate logic synthesis technique.**?** defines EXDCs of an output in a circuit as the set of primary input combinations for which that primary output is a donâĂŹt care. For the approximate circuit block under consideration here, the ADCs for a given bit of PO$_{approx}$ are the EXDCs for that output. Thus, we can transform approximate logic synthesis into traditional logic synthesis by setting the ADCs as EXDCs of an output in the approximate circuit and using these to simplify the logic cone generating that output.

4.3 Experimental Setup

The first step is to find the ADCs as a function of other inputs to the Quality function block. We hard code one of the outputs, PO_i of the approximate circuit to 1 in one quality block and to 0 in the other. We now send streams of input data (primary outputs of the original and approximate circuits, except PO_i which is hardcoded to 0/1) to the block shown below. If a particular stream when sent to the block gives an output of 0, it means that the POi has no effect on the quality function for the particular input stream and can be treated as a donâĂŹt care. Thus, we generate the ADCs in terms of the original and approximate circuit outputs.

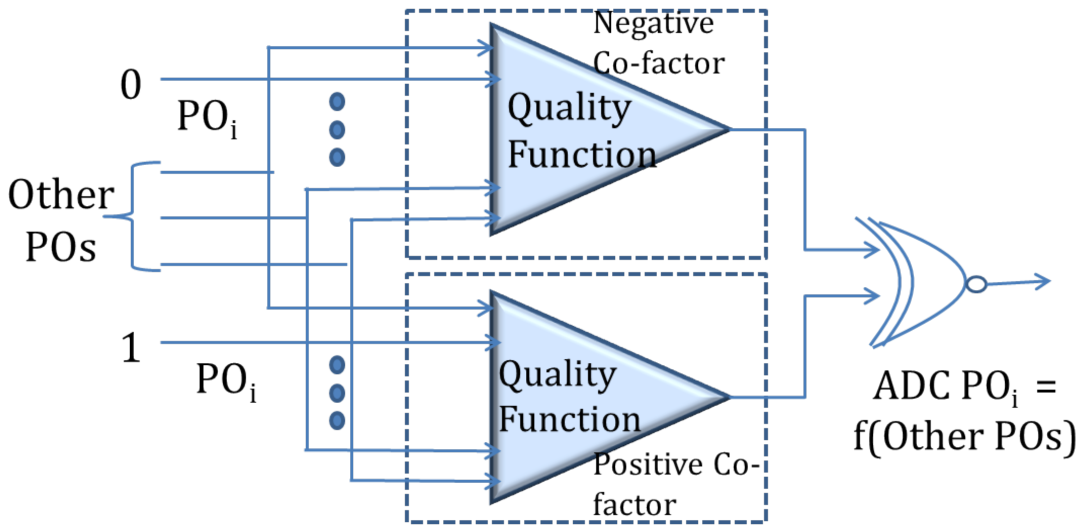


Figure 4.2: To find ADCs in terms of POs of Original and Approximate Circuits

In the next step we express the ADCs in terms of the primary inputs of the circuit. We send the stream of primary inputs to the original circuit and generate all possible outputs. We then check for the primary input combination that would produce the ADC combination obtained in the previous step. Thus for each PO_i, the ADCs in terms of primary inputs are obtained. In the next step, we specify these computed ADCs as EXDCs in the logic synthesis tool SIS and perform the traditional dont care based optimization to simplify the logic cone that generates the output bit. In the next iteration, the circuit generated above is used as the approximate circuit. We used Python scripts to automate the process of generation of ADCs and to specify them as EXDCs in the SIS file.

CHAPTER 5

Results

We present experimental results obtained by implementing the technique on various standard blocks of circuits. We then draw a comparison of the outputs obtained when the exact circuit block and the synthesized approximate circuit blocks are used in multimedia applications. The technique has been tested on a range of simple arithmetic circuits to some complex circuits from the MCNC benchmark suite.

5.1 Arithmetic and MCNC Circuits

Table5.1 and 5.2 give the list of arithmetic and MCNC benchmark circuits used to test the technique.

Name	Function	Bit Width	I/O
RCA	Ripple Carry Adder	4	8/5
RCA	Ripple Carry Adder	8	16/9
WTM	Wallace Tree Multiplier	8	16/16
CLA	Carry Look-ahead Adder	8	16/9

Table 5.1: Arithmetic Circuits used in experiments

Circuit	Gate Count	I/O	Power	Gate Area
cmb	33	16/4	311.5	82
x2	33	10/7	395.2	98
i1	38	25/16	221.2	78
cu	36	14/11	448.0	108

Table 5.2: MCNC benchmark circuits used in experiments

We first test the technique on a 4-bit Ripple Carry Adder. We set the error bound as 2, that is, in Equation 4.1, K = 2. We denote the ADCs in terms of the primary outputs of the original and approximate block according to the first step of the technique. These ADCs are then expressed in terms of the primary inputs. In the final step, these ADCs are declared as EXDCs in the BLIF representation of the circuit. We use logic synthesis tool SIS to perform do not care based logic optimization. The area and power estimation is done using the mcnc.genlib library (Appendix) that comes along with the SIS package, assuming a clock of 20 Hz and supply voltage of 5 V. The results obtained are given in the table below:

Circuit	Power	Area	Gate Count
Original	303.5 microW	61	27
Approximate	182.1 microW	48	21

Table 5.3: Area and power metrics of original and approximate 4-bit RCA

We test the technique on an 8-bit Ripple Carry Adder for various values of K. The power dissipated by the original (exact circuit) is 704.3 microW and with an area of 133. The Table5.4 below gives the power results for varying values of K.

K	Error Magnitude $(\%)$	Power(Approximate Circuit)	Relative Power
2	0.39	698.6	0.992
3	0.59	687.4	0.976
4	0.78	676.1	0.960
5	0.97	640.9	0.910
11	2.15	554.3	0.787
64	12.5	281.72	0.4

Table 5.4: Relative power vs Error Magnitude for 8-bit RCA

The Table5.5 below gives the area results for varying values of K.

K	Error Magnitude (%)	Area (Approximate Circuit)	Relative Area
2	0.39	124	0.932
3	0.59	118	0.887
4	0.78	113	0.850
5	0.97	109	0.819
11	2.15	97	0.729
64	12.5	56	0.421

Table 5.5: Relative Area vs Error Magnitude for 8-bit RCA

These results have been plotted on graphs as below. The x-axis in these graphs is in log scale.

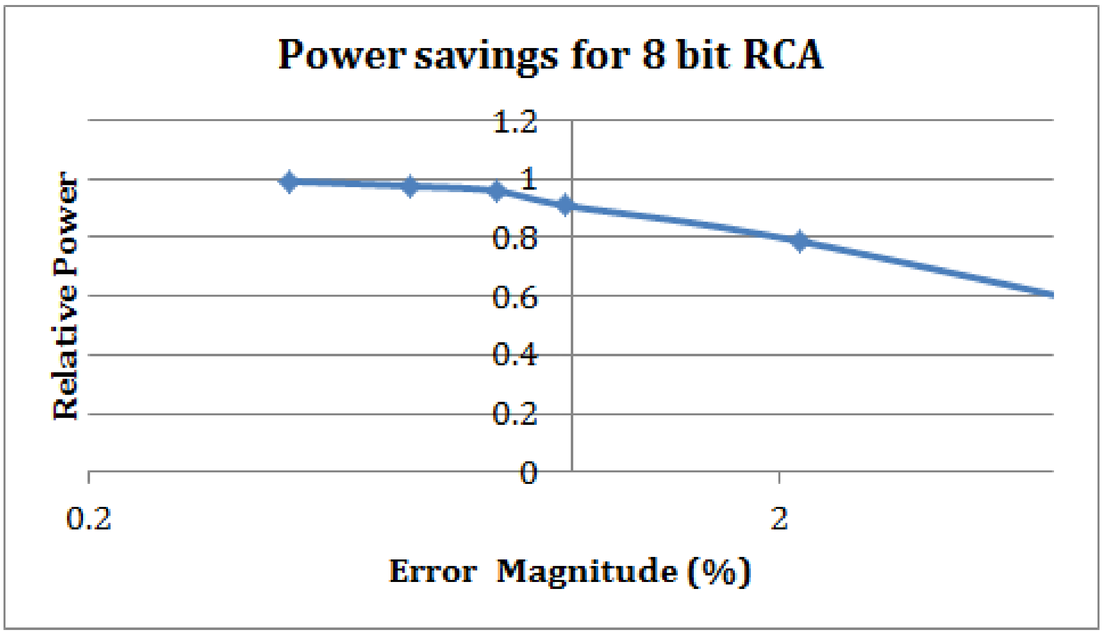


Figure 5.1: Relative power vs Error Magnitude for 8-bit RCA

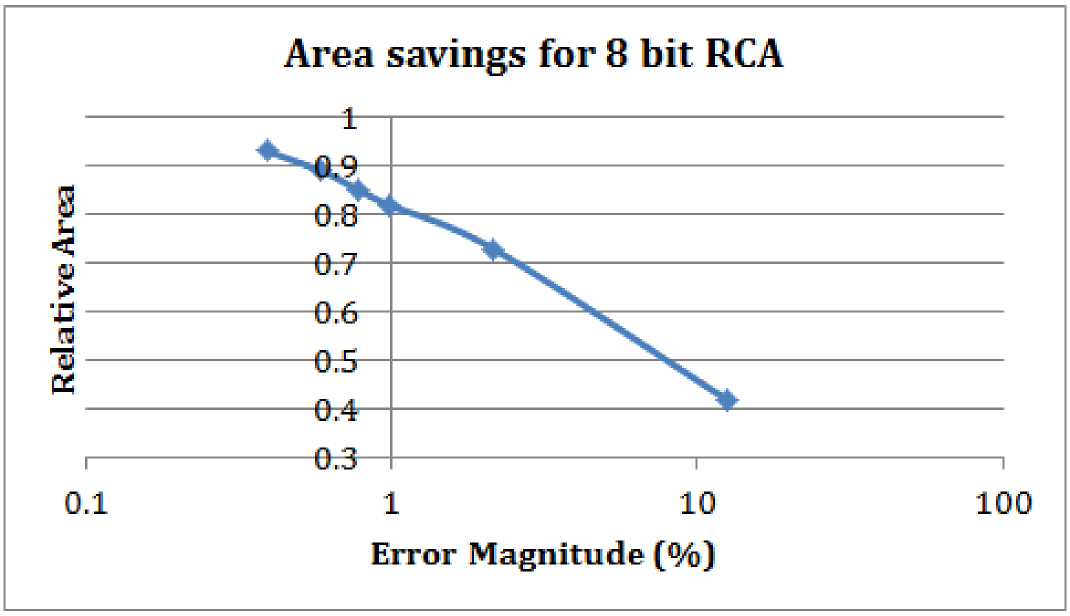


Figure 5.2: Relative Area vs Error Magnitude for 8-bit RCA

$$RelativePower = \frac{PowerdissipatedbyApproximateCircuit}{PowerdissipatedbyOriginalCircuit}$$

$$RelativeArea = \frac{AreaofApproximateCircuit}{AreaofOriginalCircuit}$$

$$Errormagnitude = \frac{Error}{MaximumOutputValue}$$

We test the technique on an 8x8 Wallace Tree multiplier and the results obtained have been plotted in Figure 5.3 and Figure 5.4 .

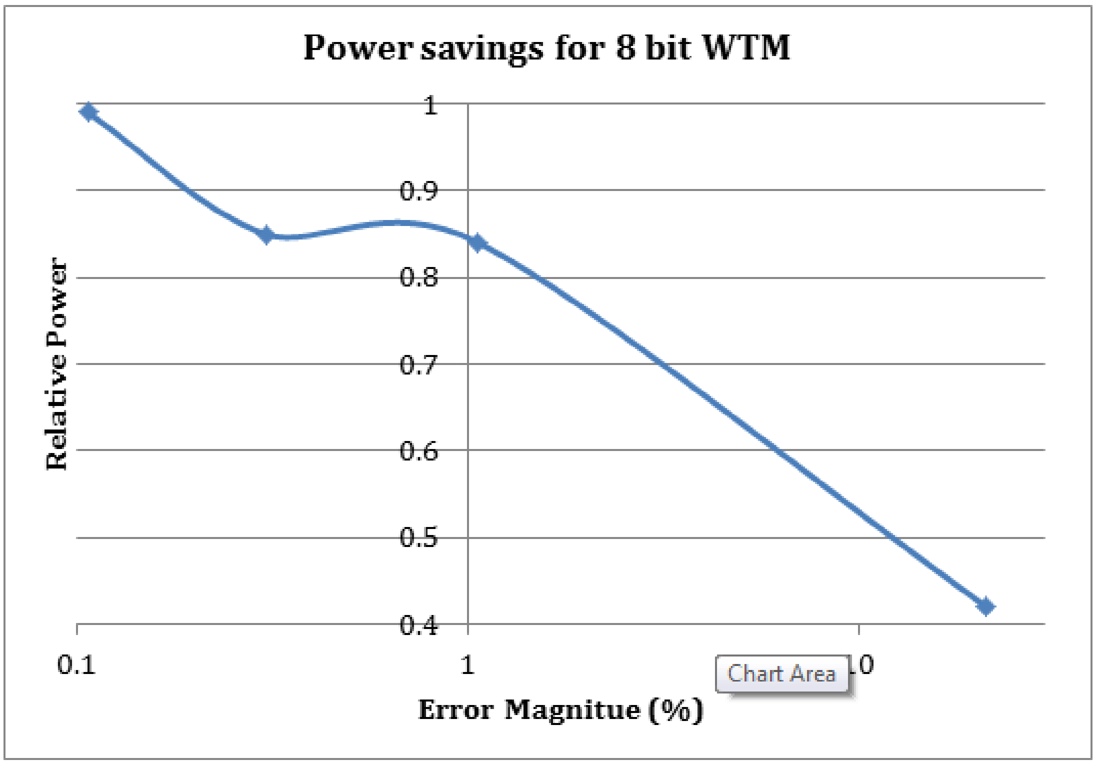


Figure 5.3: Relative power vs Error Magnitude for 8-bit WTM

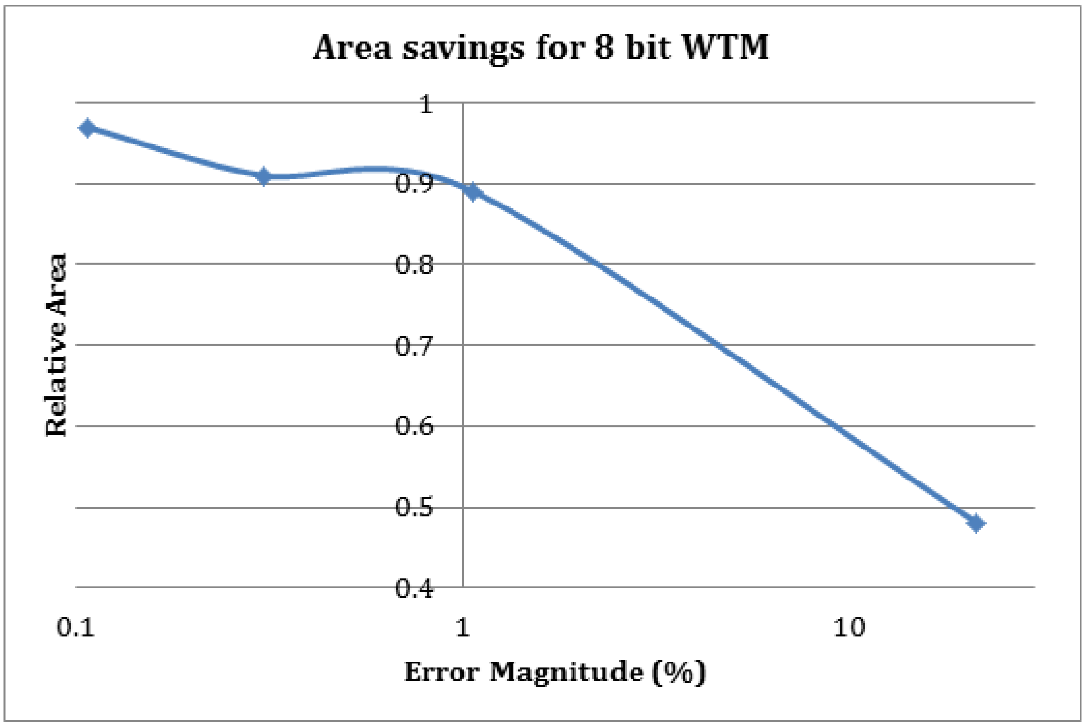


Figure 5.4: Relative area vs Error Magnitude for 8-bit WTM

The results obtained when the technique is tested on all the arithmetic and MCNC benchmark circuits listed at the beginning of this section can be seen in the graph below.

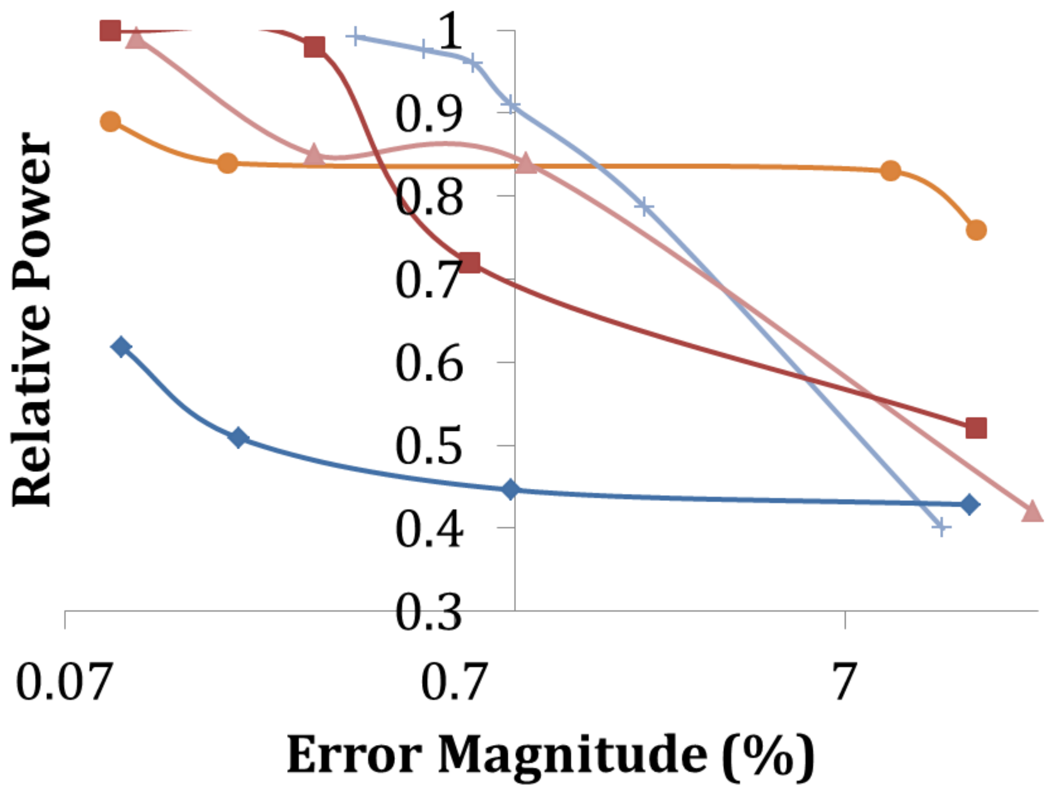


Figure 5.5: Power savings for tested circuits

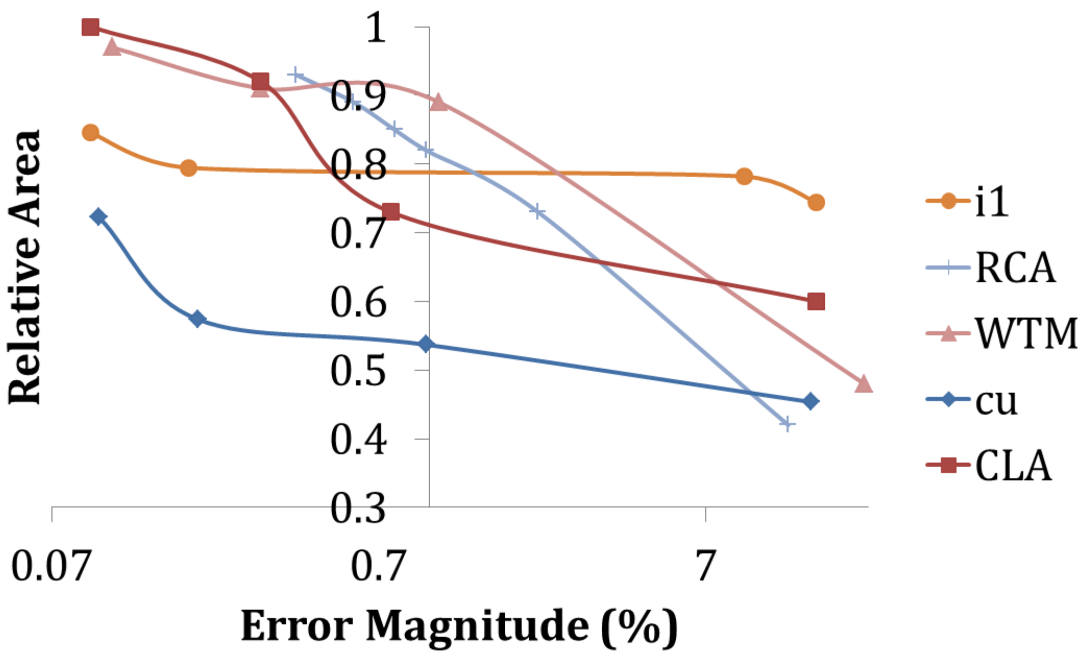


Figure 5.6: Area savings for tested circuits

From these graphs, we can see that there is an exponential decrease in the area and power initially and as we move towards larger error values, the graph tapers out. This exponential behaviour can be explained by the fact that adjacent bits have an exponential difference in their significance. Hence for the same increase in error magnitude, the incremental potential of the approximation technique is less as the value of error increases. Another point to note is that, for a given value of error, the cone of logic would generate more number for ADCs for the least significant bits than the most significant bits. Hence, approximation can be better done using the larger set of ADCs generated by the LSBs.

From the above graphs we see that the technique yields power savings in the range of 1.1X-1.2X for error bound less than 1 percent and upto 2.5X for relaxed error bounds.Similarly, area savings range from 1.1X-1.3X for error bounds upto 1 percent and up to 2.4X for relaxed error bounds.

5.2 Image Processing Applications

We test our technique is some of the image processing applications. We first test it in the process of reconstruction of images after processing them through an 8-point FFT. To construct an approximate FFT, we replace the original adders and multipliers by approximate blocks.

In Figure 5.7, consider images labelled from (a) to (f) in the clockwise order beginning from the top-left image. (a) shows the original image. (b) is the reconstructed image after processing it through the original, exact 8-point FFT. (c)-(f) are the reconstructed image after processing it through 8-point FFT with approximate arithmetic blocks with varying error bounds. The PSNR values for the corresponding error bounds have been tabulated in the following table. The table also gives the area and power savings of the approximate FFT with respect to the original FFT.

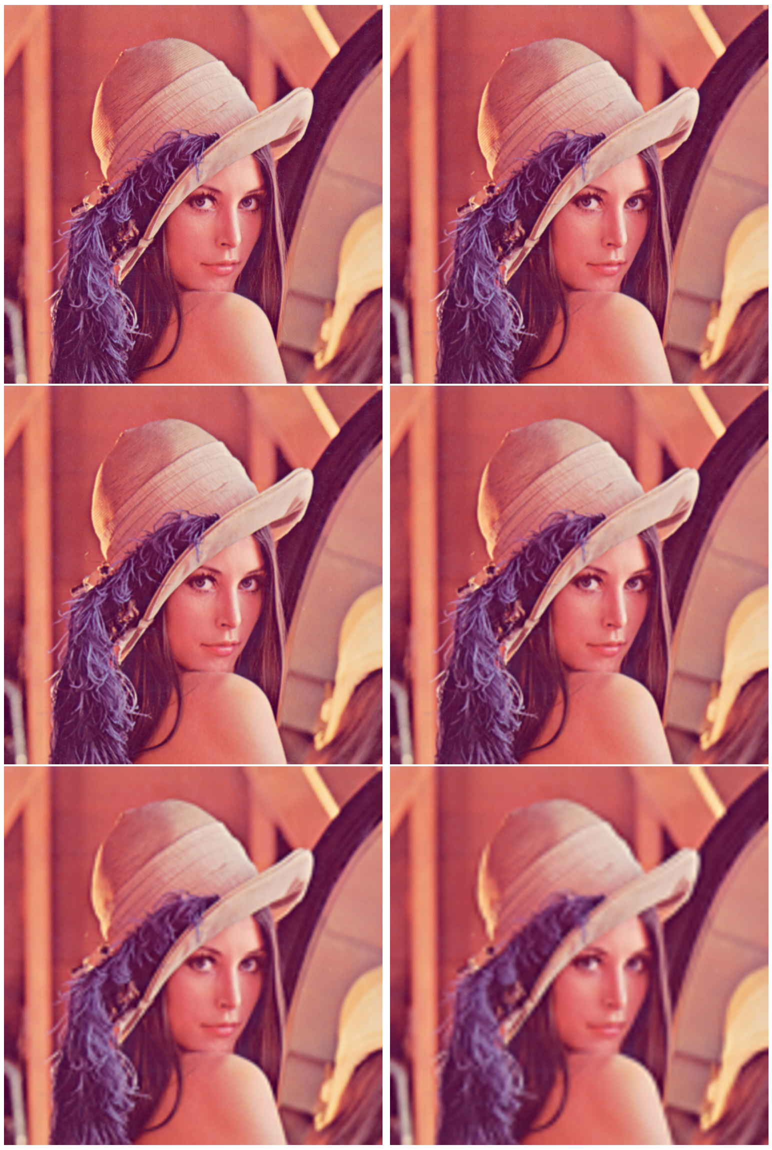


Figure 5.7: Images processed through an 8-bit FFT

$Image$	Error bound (%)	PSNR	Power Savings	Area Savings
a	-	Inf	-	-
b	0	116.88 dB	1X	1X
c	0.5	90.41 dB	1.1X	1.1X
d	1.5	78.85 dB	1.2X	1.2X
e	15	61.76 dB	1.6X	1.5X
f	26	56.51 dB	2.5X	2.4X

Table 5.6: Area and power savings in case of approximated FFT

CHAPTER 6

Conclusion

Error resilient applications provide designers a great scope for optimizing power and area of a circuit. Approximate Computing opens up to an energy aware, precision tuned hardware/software that is expected not only to reach our smart phones and tablets but go beyond that to the big servers and to the clouds. We demonstrated a technique based on Venkataramani *et al.* (2012) to functionally approximate circuits. This technique yields power savings up to 2.5X and area savings up to 2.4X for relaxed error bounds. A wide range of approximations can be applied to circuits using this technique as it uses well formulated logic synthesis tools. We demonstrated the utility of the technique across a set of standard circuits and evaluated the power and area savings. We also demonstrated the application of the synthesized approximate circuits in image processing and could see that for an acceptable loss in quality of the output, good deal of power and area savings can be obtained in such error resilient systems.

REFERENCES

1. **Allan, A.**, **D. Edenfeld**, **W. H. Joyner**, **A. B. Kahng**, **M. Rodgers**, and **Y. Zorian** (2002). 2001 technology roadmap for semiconductors. *Computer*, **35**(1), 42–53.

2. **Breuer, M. A.**, Multi-media applications and imprecise computation. *In Digital System Design, 2005. Proceedings. 8th Euromicro Conference on*. IEEE, 2005.

3. **Chakradhar, S. T.** and **A. Raghunathan**, Best-effort computing: re-thinking parallel software and hardware. *In Proceedings of the 47th Design Automation Conference*. ACM, 2010.

4. **Chandrakasan, A. P.**, **W. J. Bowhill**, and **F. Fox**, *Design of high-performance microprocessor circuits*. Wiley-IEEE press, 2000.

5. **Chen, Y.-K.**, **J. Chhugani**, **P. Dubey**, **C. J. Hughes**, **D. Kim**, **S. Kumar**, **V. W. Lee**, **A. D. Nguyen**, and **M. Smelyanskiy** (2008). Convergence of recognition, mining, and synthesis workloads and its implications. *Proceedings of the IEEE*, **96**(5), 790–807.

6. **Chu, C. C.** and **D. Wong** (1998). A matrix synthesis approach to thermal placement. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, **17**(11), 1166–1174.

7. **Ernst, D.**, **N. S. Kim**, **S. Das**, **S. Pant**, **R. Rao**, **T. Pham**, **C. Ziesler**, **D. Blaauw**, **T. Austin**, **K. Flautner**, *et al.*, Razor: A low-power pipeline based on circuit-level timing speculation. *In Microarchitecture, 2003. MICRO-36. Proceedings. 36th Annual IEEE/ACM International Symposium on*. IEEE, 2003.

8. **Gonzalez, R.**, **B. M. Gordon**, and **M. A. Horowitz** (1997). Supply and threshold voltage scaling for low power cmos. *Solid-State Circuits, IEEE Journal of*, **32**(8), 1210–1216.

9. **Gupta, V.**, **D. Mohapatra**, **S. P. Park**, **A. Raghunathan**, and **K. Roy**, Impact: imprecise adders for low-power approximate computing. *In Proceedings of the 17th*

IEEE/ACM international symposium on Low-power electronics and design. IEEE Press, 2011.

10. **Gupta, V.**, **D. Mohapatra**, **A. Raghunathan**, and **K. Roy** (2013). Low-power digital signal processing using approximate adders. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, **32**(1), 124–137.

11. **Hisamoto, D.**, **W.-C. Lee**, **J. Kedzierski**, **H. Takeuchi**, **K. Asano**, **C. Kuo**, **E. Anderson**, **T.-J. King**, **J. Bokor**, and **C. Hu** (2000). Finfet-a self-aligned double-gate mosfet scalable to 20 nm. *Electron Devices, IEEE Transactions on*, **47**(12), 2320–2325.

12. **Kao, J.**, **A. Chandrakasan**, and **D. Antoniadis**, Transistor sizing issues and tool for multi-threshold cmos technology. *In Proceedings of the 34th annual Design Automation Conference*. ACM, 1997.

13. **Kulkarni, P.**, **P. Gupta**, and **M. Ercegovac**, Trading accuracy for power with an underdesigned multiplier architecture. *In VLSI Design (VLSI Design), 2011 24th International Conference on*. IEEE, 2011.

14. **Lingamneni, A.**, **C. Enz**, **J.-L. Nagel**, **K. Palem**, and **C. Piguet**, Energy parsimonious circuit design through probabilistic pruning. *In Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011*. IEEE, 2011.

15. **Micheli, G. D.**, *Synthesis and optimization of digital circuits*. McGraw-Hill Higher Education, 1994.

16. **Pedram, M.** (1997). Design technologies for low power vlsi. *Encyclopedia of Computer Science and Technology*, **36**, 73–96.

17. **Shin, D.** and **S. K. Gupta**, A re-design technique for datapath modules in error tolerant applications. *In Asian Test Symposium, 2008. ATS'08. 17th*. IEEE, 2008.

18. **Shin, D.** and **S. K. Gupta**, Approximate logic synthesis for error tolerant applications. *In Proceedings of the Conference on Design, Automation and Test in Europe*. European Design and Automation Association, 2010.

19. **Shin, D.** and **S. K. Gupta**, A new circuit simplification method for error tolerant applications. *In Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011*. IEEE, 2011.

20. **Small, C. H.** (1994). Shrinking devices put the squeeze on system packaging. *EDN*, **39**(4), 41–54.

21. **Téllez, G. E.**, **A. Farrahi**, and **M. Sarrafzadeh**, Activity-driven clock design for low power circuits. *In Proceedings of the 1995 IEEE/ACM international conference on Computer-aided design*. IEEE Computer Society, 1995.

22. **Venkataramani, S.**, **V. K. Chippa**, **S. T. Chakradhar**, **K. Roy**, and **A. Raghunathan**, Quality programmable vector processors for approximate computing. *In Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture*. ACM, 2013.

23. **Venkataramani, S.**, **A. Sabne**, **V. Kozhikkottu**, **K. Roy**, and **A. Raghunathan**, Salsa: systematic logic synthesis of approximate circuits. *In Proceedings of the 49th Annual Design Automation Conference*. ACM, 2012.

24. **Wang, Z.** and **A. Bovik** (2006). Modern image quality assessment (synthesis lectures on image, video, and multimedia processing). *San Rafael, CA: Morgan Claypool.*

25. **Wu, Q.**, **M. Pedram**, and **X. Wu** (2000). Clock-gating and its application to low power design of sequential circuits. *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on*, **47**(3), 415–420.