CHAPTER 1

Introduction

The current research in the design of high speed VLSI architectures for real-time Digital Signal Processing (DSP) algorithms has been directed by the advances in the VLSI technology, which have provided the designers with significant impetus for porting algorithm into architecture. Many of the algorithms used in DSP and matrix arithmetic require elementary functions such as trigonometric, inverse trigonometric, logarithmic, exponential, multiplication and division functions. The realization of these functions using table lookup and polynomial approximation methods involve computationally intensive multiplications and additions/subtractions. These elementary functions can be efficiently implemented using processing elements performing vector rotations. The COordinate Rotation DIgital Computer (CORDIC) algorithm computes the elementary functions in a rather simple and elegant manner [1]. CORDIC computes two dimensional vector rotation or angle of a vector using a set of iterative equations employing simple addition and shift operations. Due to the simplicity of the operations involved, the CORDIC algorithm is well suited for VLSI implementation.

During the last 50 years of the CORDIC algorithm, a wide variety of applications have emerged. The CORDIC algorithm has received increased attention after an unified approach is proposed for its implementation [2]. Thereafter, CORDIC based computing has been the choice for scientific calculator applications, and HP-2152A co-processor, HP-9100 desktop calculator and HP-35 calculator are a few such devices based on the CORDIC algorithm [3, 4]. CORDIC arithmetic processor chip is designed and implemented to perform various functions in rotation and vectoring mode of circular, linear, and hyperbolic coordinate systems [5]. Since then, CORDIC technique has been used in many DSP applications [6] such as single chip CORDIC processor for DSP applications [7, 8], linear transformations [9]- [20], unitary transformations [21], digital filters [22]-[26], and matrix based signal processing algorithms [13], [27]-[31]. Recently, CORDIC algorithm has drawn wide attention for applications in diverse areas such as biomedical signal processing [32], neural networks [33], digital communications [34]-[36], 3D graphics [37, 38] and kinematic processing [39, 40], to mention a few.

Although CORDIC may not be the fastest technique to perform these operations, it is attractive due to its potential for efficient and low cost implementation of a large class of applications. Several modifications have been proposed in the literature for the CORDIC algorithm during the last two decades to provide high performance and low cost hardware solutions for real time computation of two dimensional vector rotation and transcendental functions [41]. In view of the requirements for high speed CORDIC, we address the design and realization of low latency and high throughput architectures.

We present the motivation to carry out this work in Section 1.1 and summarize the major contributions of this thesis in Section 1.2. Finally, the organization of the thesis is presented in Section 1.3.

1.1 Motivation

The execution speed of the CORDIC rotator is primarily governed by the iteration delay and the number of iterations. The iteration delay depends on the architecture and the technique employed to predict the new rotation direction. The number of iterations depends on the radix of the number system used in the CORDIC algorithm implementation and the method employed to eliminate iterative nature partially or completely in the computation of rotation.

1.1.1 Architecture Selection

The iteration delay of CORDIC depends on the architecture selected for its implementation. The CORDIC algorithm has traditionally been implemented using bit serial architecture with all iterations executed in the same hardware [1]. Later, the word serial architecture is proposed to reduce the computation delay at the cost of increasing hardware [2]. The speed of word serial architecture is limited by the variable shifting and carry/borrow propagate add/sub operations, rendering the conventional CORDIC implementation slow for high speed applications [2]. The variable shifting operation is avoided by unfolding the iteration process [7, 42, 43] so that each of the processing elements always perform the same iteration. Thus, the pipelined architecture with reduced iteration delay offers throughput improvement over the world serial architecture.

1.1.2 Redundant Arithmetic

The latency of parallel CORDIC and throughput of pipelined CORDIC rotator is primarily governed by the iteration delay. The iteration delay of the conventional CORDIC [1, 2] is limited by the carry/borrow propagate adder/subtractor. This bottleneck is overcome by employing redundant arithmetic [44]. While the carry

propagate adder delay had linear dependence on the word length of operands, the redundant adder/subtrator delay is independent of the word length [45, 46].

1.1.3 Computation of the Direction of Rotations

The conventional CORDIC algorithm computes the direction of rotations iteratively. The iteration delay can be decomposed into two delays, namely, the delay to predict the new rotation direction and the delay involved in the computation of rotation in the selected direction. The former delay increases with redundant arithmetic since it requires the computation of estimated value of signed digit variable and the selection function [44], [47]-[51]. The latter delay can be reduced significantly by employing redundant arithmetic to the conventional CORDIC. Although redundant arithmetic has contrasting effect on the two components of iteration delay, the overall iteration delay can be reduced by using redundant arithmetic compared to the conventional radix-2. The overall iteration delay can be further reduced by eliminating iterative nature in determining the direction of rotations completely [52]- [54].

1.1.4 Higher Radix and Parallelization of the x/y-path

Higher radix schemes have been introduced to reduce the latency or total computation time of conventional CORDIC. The radix-4 algorithm [51] is an extension of the radix-2 algorithm which halves the required number of iterations compared to that of radix-2. However, the iteration delay increases, since it takes more time to select a value from among the five rotation direction values, and to select an appropriate angle out of five elementary angles. Since the iteration delay limits the throughput of the pipelined architecture, microrotations are pipelined in two stages to increase the throughput leading to additional iterations [51]. The latency can also be reduced by parallelizing the x/y path. A few radix-2 architec-

tures are proposed in the literature to eliminate iterative nature in the x/y path partially [55] or completely [56] at the cost of scalability. Hence, the latency of higher radix CORDIC can be further reduced by eliminating the iterative nature in determining the direction of rotations and computation of rotation, or both.

1.2 Contributions of the Thesis

In this thesis, we propose architectural improvements to the design of CORDIC for reducing the iteration delay and the number of iterations. We have designed a new unfolded parallel architecture for rotational CORDIC using the redundant radix-4 arithmetic to address latency and iteration delay. Since the parallel architecture is combinational and offers low throughput, we have developed pipelined version of this architecture for achieving high throughput. These architectures are regular and scalable, making them efficient for VLSI implementation.

The latency and area of the proposed architectures are computed analytically in terms of full adder delay and area, respectively, to facilitate implementation of the architecture using any technology through the selection of appropriate logic style for full adder. In addition, the latency and area of the proposed architectures are compared with other unfolded architectures available in the literature for rotational CORDIC. The correctness of the proposed architectures is verified by FPGA implementation using Xilinx design tools and synthesis using Synopsys design compiler.

We propose a scalable architecture to achieve latency improvement by eliminating the sequential nature in the computation of direction of rotations and parallelization of the x/y path by computing the final vector as the resultant of several elementary vectors. Latency reduction of the order of ($\mathrm{O}(\log_2 n)$ + small overhead) is achieved by the proposed architecture as compared to the linear re-

duction achieved in the literature till date.

1.3 Organization of the Thesis

The rest of the thesis is organized as follows.

Chapter 2 describes the generalized CORDIC algorithm followed by the radix-2 and radix-4 CORDIC algorithms in the rotation mode of circular coordinate system.

Chapter 3 lays down the theory upon which the algorithms and architectures proposed in the thesis are based. In addition, we present the detailed analysis and comparison of algorithms and architectures available in the literature for rotational CORDIC.

Chapter 4 presents a low latency scalable parallel architecture for rotational CORDIC. In addition, the latency estimate results are presented for the proposed design and other architectures available in the literature for comparison. The implementation results of the proposed parallel architecture using programmable logic devices and Synopsys design compiler are presented.

Chapter 5 extends the precomputation of direction of rotations to design a pipelined architecture. In addition, the estimated latency improvement achieved over the other pipelined architectures is determined. Furthermore, the approximate hardware complexity comparison of pipelined architectures is presented. The functionality of the proposed pipelined architecture is verified by carrying out implementation using programmable logic devices and ASIC technology.

Chapter 6 presents a CORDIC-like architecture to reduce the number of iterations involved in the CORDIC algorithm to ($O(\log_2 n)$ + small overhead) as compared to the linear reduction achieved in the literature so far. The estimated latency and hardware of the proposed CORDIC-like fast rotator are compared

with that of unfolded parallel and pipelined architectures proposed in Chapter 4 and Chapter 5.

Chapter 7 concludes the thesis and suggests some directions for future work.