PREFACE

The discrete wavelet transform (DWT) is a multi-resolution analysis tool widely used in signal analysis such as image compression, speech analysis and pattern recognition. Several design schemes have been proposed in the last two decades for efficient implementation of 2-D DWT in VLSI system. Researchers have adopted different algorithm formulation, mapping scheme, and architectural design methods such as PA and RPA based, folded-based, parallel designs and multiplier-less designs to reduce the computational time, arithmetic or memory complexities of 2-D DWT. Among these designs parallel design offers a better utilization of memory resource which is a major component in multi-level 2-D DWT structure. Memory complexity is considered to be the design issue and overlapping data-accessing scheme is considered to reduce on-chip memory aggressively at the cost of some overhead complexity. However, the parallel design uses several data-selectors (multiplexors and de-multiplexors) apart from multiplier, adder and memory elements. These data-selectors are used for time-multiplexing data sequences to improve the resource utilization of the hardware design while performing down-sample filter computation. The data-selector complexity depends on the block-size which is usually very large and mapping algorithm. The data-selector complexity affects the area-delay efficiency of the parallel design substantially. However, the data-selector complexity somehow is overlooked in the existing parallel designs. Using appropriate algorithm formulation and architecture design, the data-selector complexity can be avoided completely to improve the hardware efficiency of the parallel designs without considering aggressive memory optimization with overhead complexity. Also, it is observed that that the parallel design for large block size involves hundreds of multipliers which contribute almost comparable amount area as the on-chip memory unit. Use of an optimized multiplier design could improve the area-delay efficiency of the parallel design substantially. In chapter-3 the data-access schemes used in the existing parallel designs are studied to identify the data-multiplexing instances in multi-level lifting 2-D DWT computation. Based on this study, a novel data-access scheme is formulated to avoid data-multiplexing which is common in the existing parallel architectures. A block formulation based on the proposed data-access scheme is presented for vector computation of multi-level lifting 2-D DWT. A generic processing unit design is presented using the proposed block-formulation which is free from data-selectors. The input-block size is used to configure the generic design to perform lifting 2-D DWT computation of different decomposition levels. A regular and modular

parallel architecture is derived using the generic processing unit design. The proposed parallel architecture is easily scalable for higher block-sizes as well as higher DWT levels without sacrificing its circuit regularity and modularity. This is an important feature of the proposed architecture. In **Chapter-4**, a review of radix-4 and radix-8 Booth multiplication algorithms using different encoding scheme such as anti-symmetric product coding (APC) symmetric, symmetric product coding (SPC) and symmetric anti-symmetric product coding (SAPC) is presented. A detail complexity analysis of radix-4 and radix-8 Booth multiplication is presented to study the design advantages of multiplication algorithms using different Booth encoding schemes. Based on the complexity study, a design strategy is considered to find an efficient design radix-8 Booth multiplier. **Chapter-5** discusses different types of multiplications encountered in DSP application. These multiplication operations are broadly divided into two types such as generic and constant, where the constant multiplication further divided as generic fixed constant the fixed-constant multiplication. Booth algorithm offers a design for generic multiplier. The generic radix-8 multiplier creates some redundant logic when the multiplier is used for generic-constant multiplication and the multiplier result is post truncated in case of fixed-width implementation. These redundant logic operations are identified and removed to obtain an optimized design for post-truncated fixed-width radix-8 Booth multiplier. The block-based lifting 2-D DWT algorithm uses a large number of constant multiplier with one common operand. This is an interesting property and can be used to save area complexity of the hardware design. The block lifting 2-D DWT algorithm is presented using scalar-vector product to take advantage of common multiplier operand feature. The parallel design of Chapter 3 is presented in a modified form in Chapter-5 using scalar-vector product and the proposed radix-8 generic-constant fixed-width multiplier to demonstrate the effectiveness of proposed scheme. The lifting constants are fixed. A fixed constant multiplier can implemented using look-up-table (LUT) to save area as well power since ROM LUT involves less area and consumes less power than the partial product selector unit of generic fixed constant multiplier which is implemented using multiplexer. To take advantage of this, **Chapter 6,** presets LUT based sign multiplier design where different types of encoding schemes are considered to improve the design efficiency. The proposed LUT multiplier is used in the proposed lifting 2-D DWT structures which involves sign multiplication. The proposed LUT-multiplier offers an efficient hardware design for high-speed implementation of lifting 2-D DWT.