HARDWARE IMPLEMENTATION AND FIXED-POINT ERROR ANALYSIS OF THE SYMMETRIC WAVELET TRANSFORM

CARLOS L. BETANCOURT

Electrical and Computer Engineering Department

APPROVED:

Dr. Bryan Usevitch, Chair

Dr. Sergio Cabrera

Dr. Soheil Nazarian

Associate Vice President
for Graduate Studies

Para mis padres, mi esposa y mi hermano,

quienes siempre me han brindado su amor y

su apoyo incondicional.

HARDWARE IMPLEMENTATION AND FIXED-POINT ERROR ANALYSIS

OF THE SYMMETRIC WAVELET TRANSFORM

by

CARLOS L. BETANCOURT, B.S.E.E.

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fullfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

Electrical and Computer Engineering Department

THE UNIVERSITY OF TEXAS AT EL PASO

May, 1999

Acknowledgements

I would like to give special thanks to Dr. Bryan Usevitch and Dr. Sergio Cabrera for their guidance and support throughout my career, and to Dr. Soheil Nazarian for being part of my Supervising Committee. I would also like to express my gratitude to Alexander G. Contreras for his contributions on the preliminary work of this research, and to Victor Villalva for his assistance in configuring and testing the designs on actual chips. Finally, I would like to add a special note of appreciation to all the members of the Signal Processing and Communications Affinity group for their support and friendship.

This work was supported in part by the NASA Faculty Awards for Research (FAR) contract number 961119. This thesis was submitted to my Supervising Committee on March 11, 1999.

Abstract

An implementation of the symmetric wavelet transform (SWT) to be used as a pre-processing stage of a data compression system is presented. A rapid prototyping approach is adopted, in which the designs are first developed using the Signal Processing WorkSystem® (SPW®) and then ported onto an Altera® Field Programmable Gate Array (FPGA). The designs comprise a perfect reconstruction (PR) filter bank that uses the Daubechies-9/7 coefficients. The implementation includes single-multiplier filters and hardware extension circuits that allow for input signals of length 256 and provide four levels of decomposition. The fixed-point arithmetic effects are studied to predict reconstruction signal-to-noise ratios (SNR's) based on the filter word lengths. The filter coefficients are scaled to maintain dynamic range in a dyadic-tree decomposition, which prevents perfect alias cancellation (PAC). The main contribution of this thesis is the derivation of constraints to enforce PAC which, although not proven to maximize SNR, makes the error analysis more tractable. The theoretical results are compared with simulation results for twelve one-dimensional input signals obtained from a raster scan of 128×128-pixel Synthetic Aperture Radar (SAR) images. It is shown that on the average, when the outputs are not requantized to the same number of bits as the input, SNR's can be predicted to within 1.5% accuracy. For the case where the output is requantized, a statistical approach is proposed for selecting the filter word lengths which will produce PR to a desired standard deviation that can be determined experimentally.

Contents

List of Figures

Chapter 1

Introduction

1.1 Data Compression

Data Compression is so common today that we make use of it almost daily during our lives. Electronic devices such as modems, fax machines, personal computers, laser printers, and many more often make use of data compression algorithms before transmitting or storing information. Data compression is commonly referred to as the "art" or "science" of representing information in the most compact form, eliminating as much redundancy as possible [1].

One of the earliest examples of data compression is Morse code. Developed in the mid-19th century by Samuel Morse, this code makes use of dots and lines to represent letters of the alphabet. Morse noticed that some letters appeared more often in text than others. In view of this, he assigned shorter representations to letters such as a and e, which appeared more often in text, and longer representations to letters such as q and j, which appeared less often in text. By doing this, he reduced the average time required to transmit a message [1].

The need for data compression is rapidly increasing as databases continue to grow and new technology becomes part of our lives. For example, the fingerprint

database of the US Federal Bureau of Investigation (FBI) grew up to 114 million fingerprint cards when last counted. In response to the need for faster background checking, the FBI decided to convert this database from paper to digital electronic format. However, digitizing the current database without compression would produce about 1,140 terabytes of information. To reduce the digital fingerprint file sizes, the FBI implemented the Wavelet/Scalar Quantization (WSQ) compression standard, which can achieve acceptable image-quality compression ratios of around 20:1 [2]. As another example, consider the transmission of digital television signals. If we wanted to transmit High Definition Television (HDTV), we would require data rates in the order of 884 Mbps without compression. This data rate would occupy a bandwidth of about 220 MHz. But with compression, HDTV data rates can be reduced to less than 20 Mbps, requiring a bandwidth of only about 6 MHz, which is the bandwidth allocated to transmit analog television in the US [1]. These two examples illustrate the increasing necessity of data compression, and just like these two, there are many other examples of applications that need some sort of data compression scheme to become practical.

The scope of this research lies within the scope of data compression. A typical data compression algorithm is composed of three stages: Pre-Processing, lossy, and lossless (see Fig. 1.1). The lossless stage is commonly comprised of an entropy coder such as Huffman or Arithmetic. As implied, the lossless stage preserves all the information after compression. The lossy stage, on the other hand, takes advantage of the structure present in the information to discard parts of it that are redundant. A quantizer is usually utilized to perform lossy compression and it can be scalar, vector, or some other type of quantizer. The pre-processing stage usually involves taking a transformation of the data such as a Discrete Cosine Transform (DCT), a Symmetric Wavelet Transform (SWT), or some other transform. Whichever transform is chosen, the objective of the pre-processing stage is to "massage" the information so that its

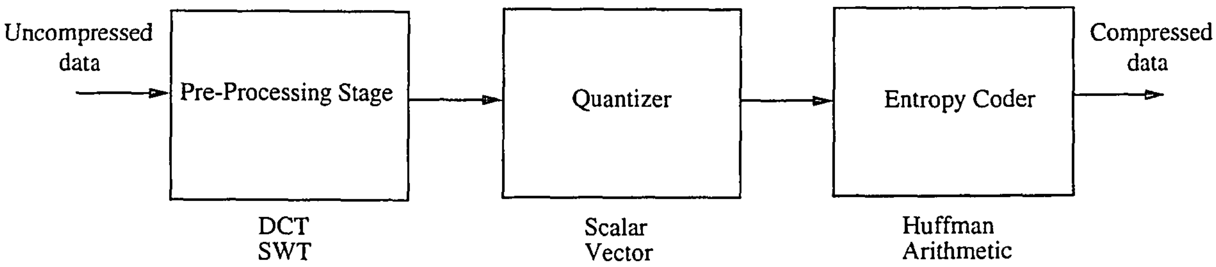

Figure 1.1: Typical stages of a data compression algorithm. Examples are provided under each box.

structure gets maximally exploited by the lossy stage. The *Symmetric Wavelet Transform* (SWT) as a pre-processing stage of a data compression algorithm is the main focus of this thesis.

1.2 Rapid Prototyping and Research Objectives

Rapid prototyping has become more popular as computer software technology advances. Computer Aided Design (CAD) tools provide a friendly environment for designing and simulating electronic systems. CAD tools include functions that are commonly used in certain applications. For example, a CAD tool might include a "filter" block for Digital Signal Processing (DSP) applications. However, not only is a friendly environment what a designer looks for, but also a way of reproducing the design in actual hardware. This is what rapid prototyping is all about, to be able to design and simulate systems using a CAD tool and to be able to port such systems onto real hardware.

As further research is conducted in configurable computing, platforms based on Field Programmable Gate Arrays (FPGA's) have become more popular. The notion of being able to produce an FPGA-based prototype in a relatively short period of time and with minimal cost is appealing. It is the flexibility provided by FPGA's that make possible an assessment of a system functionality previous

to delivering the final product, possibly an Application-Specific Integrated Circuit (ASIC). As opposed to ASIC's, FPGA's are flexible enough to track any changes made in a system by providing the designer with reprogrammable logic. The trade-offs associated with FPGA design are high power requirements, low clock speed, and, in applications where reconfigurable computing is necessary, long programming times. However, current research that aims at improving the performance of FPGA's over these liabilities motivates the development of algorithms in hardware using the idea of rapid prototyping [3].

The objective of this research is to design and implement a SWT chip for data compression applications using rapid prototyping. The CAD tool to be used for this project is the Signal Processing WorkSystem® (SPW®) by Cadence. SPW allows for block design and simulation of DSP systems in floating-point. Then, the design can be converted into fixed-point in order to generate the Hardware Description Language (HDL) code [4]. After generating the HDL code, one can compile it and synthesize it using Synergy, another piece of software by Cadence. Finally, the design can be down-loaded into an Altera® FPGA. This process is illustrated in Fig. 1.2.

This research includes the characterization of the SWT hardware implementation. In theory, the SWT provides perfect reconstruction (PR), which means that the output signal is equal to the input signal except maybe for a scaling factor and a time delay [5, 6, 7, 8]. When implementing the SWT in hardware, however, finite word length arithmetic introduces error that prevents us from obtaining PR. How much error is expected at the output is a question that is addressed in this research. It is our objective to perform a fixed-point error analysis on the hardware implementation of the SWT in order to predict average reconstruction signal-to-noise ratios (SNR's) at the output of the chip.

In what follows, Chapter 2 introduces the concepts of wavelets, multiresolution techniques, and the conditions for orthogonality and biorthogonality in PR filter

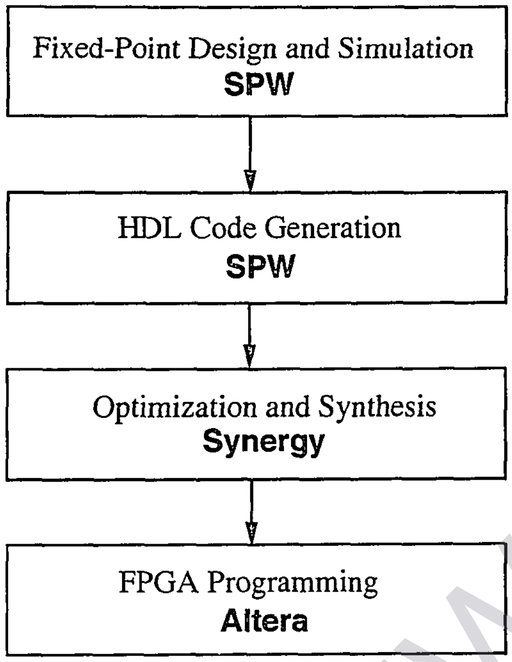

Figure 1.2: Procedure for rapid prototyping.

banks. Chapter 3 deals with the problem of expansion effects and introduces symmetric extension transforms (SET's) as a solution to the problem. Chapter 4 gives the details about the hardware implementation of the SWT and ISWT systems. Chapter 5 presents a fixed-point error analysis of two-channel PR filter banks to determine the expected reconstruction SNR's for different filter word lengths. Chapter 6 gives conclusions and future work.

Chapter 2

The Wavelet Transform and Filter Banks

2.1 Wavelet Analysis

It has been well documented in the current literature (see for example [6, 9]), that wavelet analysis represents an alternative to Fourier analysis when dealing with non-stationary signals. The local behavior of a signal can be better approximated with a relatively small number of non-zero coefficients when doing wavelet analysis. Fourier analysis, on the other hand, often requires more coefficients to be able to recapture local details. By having more coefficients that are close to zero, one can simply discard these and keep the ones above a certain threshold. Compression can be achieved this way because we are able to represent (or accurately approximate) a signal using less coefficients. Although this is generally true, it is important to point out that wavelet analysis is not always the answer. For example, if a signal is composed of a few sinusoids, then performing Fourier analysis on that signal would actually produce less coefficients than wavelet analysis.

The wavelet transform is defined as [6]

$$X(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} h^* \left(\frac{t-b}{a} \right) x(t)\, dt,$$

where the basis functions or wavelets are of the form

$$h_{a,b}(t) = \frac{1}{\sqrt{a}} h \left(\frac{t-b}{a} \right) \qquad a, b \text{ real}; a > 0. \tag{2.1}$$

The function $h(t)$ in (2.1) is often called the "mother wavelet." By stretching/contracting (varying a) or by shifting (varying b) this mother wavelet one obtains the wavelet basis functions. Wavelets do not extend infinitely in time. Wavelets are time-limited and that is why it is often said that wavelets have "compact support."

When the parameters a and b in (2.1) are continuous, the set of basis functions is highly redundant. A discretization of these parameters that will produce an orthonormal set of basis functions is highly desired. Of particular interest is the discretization on a dyadic grid, which gives basis wavelet functions of the form

$$h_{j,k}(t) = 2^{-j/2} h(2^{-j}t - k),$$

where j and k are integers. Such a set of orthonormal basis functions is possible to construct [6].

2.2 Multiresolution and Filter Banks

Wavelets have become very popular because of their ability to decompose signals at different "resolutions." The idea of multiresolution has found a lot of applications, especially in subband coding where a signal is decomposed into bands of frequency for compression and transmission [10, 11]. It is here where filter banks get into the picture of multiresolution since by iterating digital filters one can achieve different levels of resolution. Filter banks are thus very closely related to wavelet analysis

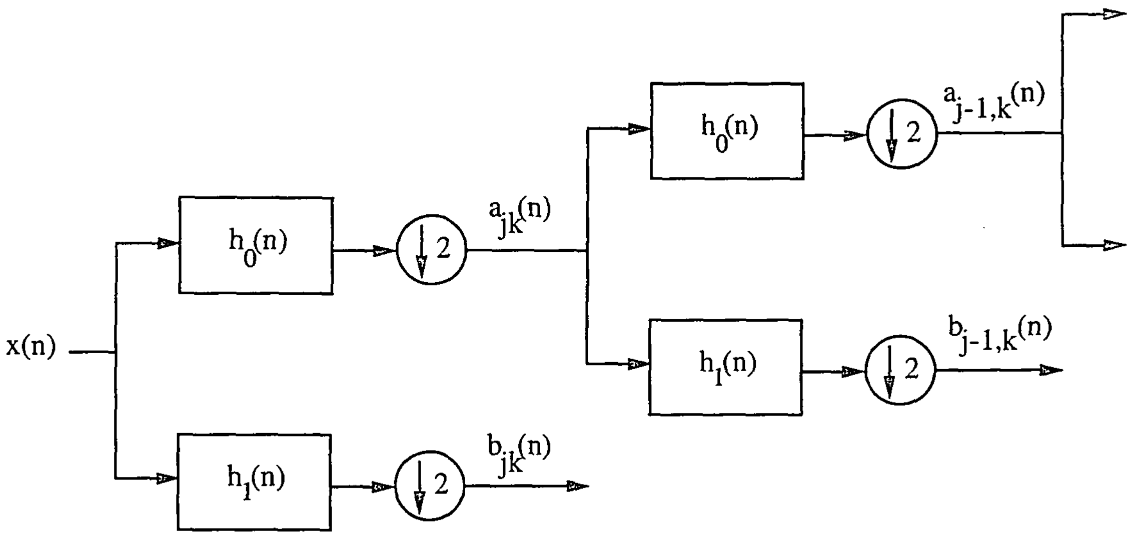

Figure 2.1: Dyadic tree-structured filter bank for analysis stage.

and multiresolution techniques, and they represent a practical way of applying such concepts to image compression.

In discrete-time wavelet analysis, a signal $x(n)$ is decomposed into a finite number of resolutions using an analysis filter bank (see Fig. 2.1). At a given level of resolution j, the scaling coefficients $a_{jk}(n)$ provide the local averages (usually containing most of the energy), and the wavelet coefficients $b_{jk}(n)$ provide the local differences. The "smooth" signal given by the scaling coefficients and the details added by the wavelet coefficients combine at level j to produce a finer signal at level of resolution $j+1$. The scaling coefficients are obtained by several iterations of a lowpass filter $h_0(n)$, and the wavelet coefficients follow from the scaling coefficients by just one application of a highpass filter $h_1(n)$.

2.3 Perfect Reconstruction Filter Banks

As we have pointed out, the purpose of transforming the input data is to "massage" it so that the lossy stage can maximize compression by taking advantage of

the structure contained in the data (see Fig. 1.1). A transformation can be of three types [8]:

- Lossless (orthogonal) transforms
- Invertible (biorthogonal) transforms
- Lossy transforms

Lossless transforms preserve energy and they are like a rotation of a vector, preserving its length but changing its angle. Lossless transforms lead to orthogonal and unitary matrices. The Fourier transform and the DCT are examples of lossless transforms. Invertible transforms change the length and the angle of a vector, thereby not preserving its energy. However, they are completely invertible. Invertible transforms are also called biorthogonal. They lead to invertible matrices and give perfect reconstruction (PR). When small components in the data are destroyed during the transformation, then the transform becomes irreversible and is called lossy. In this section, the conditions for biorthogonality and orthogonality in filter banks are discussed.

Our discussion is based on the two-channel filter bank shown in Fig. 2.2. Notice that the filter bank is a multirate system since down-sampling and up-sampling operations are involved. To obtain the input-output relationship of the two-channel filter bank, consider a signal $u(n)$ that is down-sampled by two to give $v(n)$. The input-output relationship between $u(n)$ and $v(n)$ in the Z-domain is given by

$$V(z) = \frac{1}{2}[U(z^{1/2}) + U(-z^{1/2})]. \tag{2.2}$$

Now consider another signal $p(n)$ that is up-sampled by two to give $q(n)$. This gives the relationship

$$Q(z) = P(z^2). \tag{2.3}$$

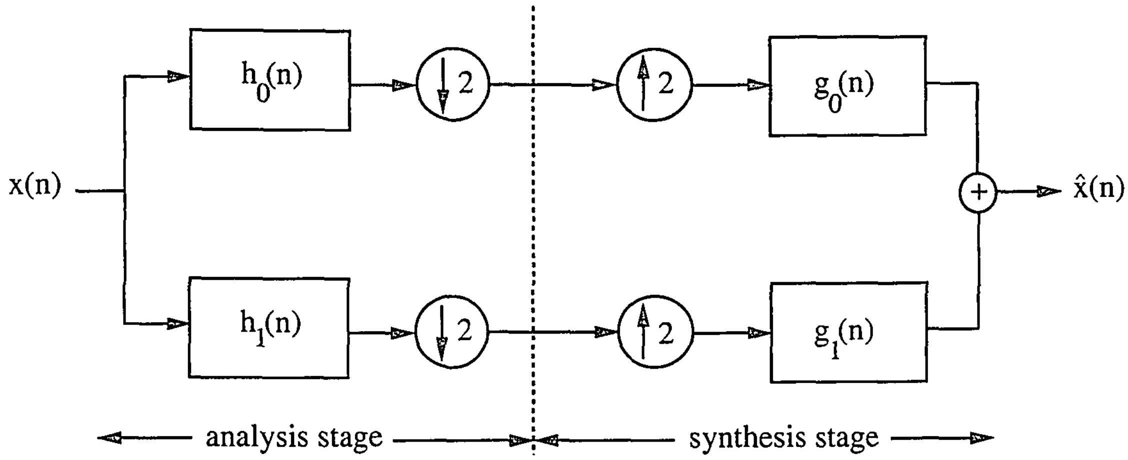

Figure 2.2: Two-Channel filter bank.

Using Equations (2.2) and (2.3) we can obtain the input-output relationship for the two-channel filter bank:

$$\begin{aligned}\hat{X}(z) &= \frac{1}{2}[H_0(z)G_0(z) + H_1(z)G_1(z)]X(z) \\ &\quad + \frac{1}{2}[H_0(-z)G_0(z) + H_1(-z)G_1(z)]X(-z) \\ &= T(z)X(z) + S(z)X(-z),\end{aligned}$$

where

$$T(z) = \frac{1}{2}[H_0(z)G_0(z) + H_1(z)G_1(z)] \tag{2.4}$$

$$S(z) = \frac{1}{2}[H_0(-z)G_0(z) + H_1(-z)G_1(z)]. \tag{2.5}$$

The signal $X(-z)$ is an aliased version of $X(z)$ and is thus undesired. In order to obtain PR we must have

$$T(z) = Kz^{-n_0}, \qquad \text{where } K \text{ and } n_0 \text{ are constants; and} \tag{2.6}$$

$$S(z) = 0, \qquad \forall z. \tag{2.7}$$

To get rid of $X(-z)$ we can impose the condition

$$\frac{G_0(z)}{G_1(z)} = -\frac{H_1(-z)}{H_0(-z)},$$

where a typical solution is [5, 6]

$$\begin{aligned} G_0(z) &= H_1(-z) \\ G_1(z) &= -H_0(-z). \end{aligned}$$

This solution enforces the condition in (2.7), which leaves us with

$$\hat{X}(z) = T(z)X(z),$$

where $T(z)$ is defined in (2.4).

To get PR, we must meet the condition in (2.6). In the classic quadrature mirror filter (QMF) solution [5], the analysis filters are related by

$$H_1(z) = H_0(-z),$$

which gives

$$T(z) = H_0^2(z) - H_1^2(z). \tag{2.8}$$

The problem with (2.8) is that there are only two cases that satisfy (2.6) exactly. One is the case where the analysis filters are ideal half-band filters (i.e. infinitely long). The other is where the order of the filters is one or less. A typical example is the Haar filter set given by

$$\begin{aligned} H_0(z) &= 1 + z^{-1} \\ H_1(z) &= 1 - z^{-1}. \end{aligned}$$

Those two cases are not practical because infinitely long filters are not realizable and because first-order filters do not have the same frequency resolving power as higher-order filters. People have worked, however, on approximating (2.8) to give (2.6). A good example is the Johnston QMF family of filters [12], which still does not produce PR but approximates it very closely.

A special case of PR filter banks is *orthogonality.* The orthogonal or conjugate quadrature filter (CQF) solution is achieved by the condition [5, 6, 13]

$$H_1(z) = -z^{-N}H_0(-z^{-1}),$$

where N is odd and $H_0(z)$ is no longer assumed to be linear phase. With this condition $T(z)$ becomes

$$\begin{aligned} T(z) &= \frac{1}{2}[H_0(z)H_0(z^{-1}) + H_0(-z)H_0(-z^{-1})]z^{-N} \\ &= \frac{1}{2}[R_0(z) + R_0(-z)]z^{-N}, \end{aligned} \tag{2.9}$$

where $R_0(z)$ is the autocorrelation sequence of $H_0(z)$ given by

$$R_0(z) = H_0(z)H_0(z^{-1}).$$

In order for Equation (2.9) to meet the condition in (2.6), $r_0(n)$ (the inverse Z-transform of $R_0(z)$) must satisfy

$$r_0(n) = 0, \qquad \text{for } n \text{ even except at } n = 0.$$

It has been proven, nontheless, that the only orthogonal real FIR filter bank that has linear phase is of order one [6], which corresponds to the Haar filter set and its shifted versions:

$$\begin{aligned} H_0(z) &= z^{-l} + z^{-l-2n-1} \\ H_1(z) &= (z^{-l} - z^{-l-2n-1})z^{-2n}, \end{aligned}$$

where l and n are integers.

As we have seen, the QMF solution of filter banks approximates but does not give PR, and the only two cases that do give PR are not practical. On the other hand, orthogonal (CQF) filter banks do not have linear phase except for one case, the Haar filter bank. Linear phase is desirable because then several filters can be

Table 2.1: Common biorthogonal filter sets.

13/11 filter coef.		9/7 filter coef.		5/3 filter coef.		filter
h_0	g_0	h_0	g_0	h_0	g_0	index
0.767245	0.832848	0.852699	0.788486	1.060660	0.707107	0
0.383269	0.448109	0.377402	0.418092	0.353553	0.353553	-1, 1
-0.68878	-0.069163	-0.110624	-0.040689	-0.176777		-2, 2
-0.033475	-0.108737	-0.023849	-0.064539			-3, 3
0.047282	0.006292	0.037828				-4, 4
0.003759	0.014182					-5, 5
-0.008473						-6, 6

cascaded in a dyadic tree without the need for phase compensation [11]. When linear phase filter coefficients are desired, one has to give up orthogonality. Consequently, biorthogonal filter banks that satisfy (2.6) and (2.7) exactly and yet provide linear phase have become the solution to many applications that make use of the analysis/synthesis system in Fig. 2.2. Examples of common biorthogonal filter sets are shown in Table 2.1 [14].

In this chapter, we have introduced the concepts of wavelet transform and filter banks. The next chapter deals with expansion effects when filtering finite length inputs. As we shall see, expansion effects are undesirable and symmetric extension transforms (SET's) represent a viable solution to the problem.