CMOS AREA IMAGE SENSORS WITH PIXEL LEVEL A/D CONVERSION

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Boyd Fowler
October 1995

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Abbas El Gamal
(Principal Adviser)

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Bruce Wooley

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Michael Godfrey

Approved for the University Committee on Graduate Studies:

Abstract

This thesis describes the first known CMOS area image sensor with pixel-level analog to digital conversion. The A/D conversion is performed using a one-bit first-order sigma delta modulator at each pixel. The sensor outputs digital data and allows for both programmable quantization and region of interest windowing. Two pixel level A/D conversion circuits are described and analyzed. Experimental results are presented for these pixel circuits. Results show that our sensor achieves low power dissipation less than 50nW per pixel and a dynamic range greater than 80dB. The sigma delta modulated pixel data must be decimated in order to recover pixel data in a base two format. Linear and nonlinear decimation techniques are compared with respect to SNR, computational complexity, and required silicon area using both still and video image data. We also show that our sensor can have lower temporal distortion than existing sensors such as CCDs.

This thesis also describes a JBIG compliant, quadtree based, lossless image compression algorithm. This algorithm is intended for both sigma delta modulated and binary image sensors. In terms of the number of arithmetic coding operations required to code an image, this algorithm is significantly faster than previous JBIG algorithms. Based on this criterion, our algorithm achieves an average speed increase of more than 9 times with only a 5% decrease in compression when tested on the eight CCITT bi-level test images and compared against the basic non-progressive JBIG algorithm. The fastest JBIG algorithm that we know of, using "PRES" resolution reduction and progressive buildup, achieved an average speed increase of less than 6 times with a 7% decrease in compression, under the same conditions.

Acknowledgements

I would like to first and foremost thank my loving wife **Roe Turco-Fowler** for her support and encouragement, especially during the writing of this thesis. I also want to thank my parents **Albert and Shirely Fowler**, my sisters **Mary Husted** and **Kimerly Fowler**, my aunt **Ruth Fowler**, and my cousin **Chris Fowler** for there support and encouragement.

Throughout the years many people have directly and indirectly helped me achieve this goal. I would like to thank them all, but there are some people who need special recognition. First I wish to thank **Carolyn Luckenbach** and her son **Sidney Luckenbach** for saving my life and supporting me when I really needed them. I thank my aunt **Ruth Fowler** for opening her house and her heart to me when I came to Stanford. Without her I would not have come to Stanford.

I thank **Abbas El Gamal** for his patience and direction. I thank **Michael Godfrey** for helping develop my creativity, and for building an experimental VLSI laboratory within Stanford's information system lab, where most of this research was conducted. I thank **Bruce Wooley** for many enlightening circuit discussions. I thank my orals committee, **Abbas El Gamal**, **Bruce Wooley**, **Michael Godfrey**, and **James Plummer** for taking the time to listen. I also thank my reading committee **Abbas El Gamal**, **Bruce Wooley**, and **Michael Godfrey** for their comments and suggestions concerning this thesis.

My time at Stanford has been intellectually and socially enriched by the friends I have made. I thank all of my office mates, **Ivo Dobbelaere**, **Dana How**, **Sanko Lan**, **David D.X. Yang**, and **Avi Ziv** for simulation conversation and the beer. I also thank **Neal Bhadkamkar**, **Jim Burr**, and **Steve Piche** for many thought

provoking discussions.

Finally, I wish to acknowledge the financial support of Texas Instruments, MSI Semiconductor/Larry Matheney, and CIS. I would also like to thank HP and Intel for their generous equipment donations, and MOSIS for their semiconductor fabrication.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In many applications, including machine vision, surveillance, video conferencing and digital cameras, it is desirable to integrate A/D conversion with an area image sensor. Such integration lowers power consumption, improves reliability, and reduces system cost. To realize these benefits, we must consider the trade-offs introduced by integration. On the one hand, integration allows parallelism which, if properly exploited, can reduce system power and speed up data conversion. On the other hand, the A/D conversion circuitry cannot be too complex because of limited chip area. Moreover, most of the chip area on an area image sensor must be occupied by photodetectors, leaving only a small proportion of the chip for active circuitry.

Today most digital imaging systems are comprised of many descrete units. A typical imaging system is shown in Figure 1.1. This system is composed of a clock driver, a CCD image sensor [1, 69] a high speed A/D converter [40], RAM, and one or more ASICs. The clock driver supplies the control signals for the CCD. The CCD image sensor converts light intensity into an analog signal. The analog signal is digitized using the high speed A/D converter, and the digitized data is stored in the RAM. The ASICs are used to manage the system and perform any necessary signal processing or data compression. One might naively think that a single chip digital sensor could be built by integrating these circuits on a single silicon substrate, but this is presently not possible because each circuit requires a different process technology. Other limitations imposed by this system include high power dissipation, $\approx$ 10 watts,

and large size. This limits the sensor's utility in portable applications.

In order to integrate a digital imaging system on a chip, all of the blocks shown in Figure 1.1 (perhaps with the exception of the RAM) must be constructed using a common process technology. This can be achieved by using a standard digital CMOS process. Figure 1.2 shows a block diagram of such a system. It is composed of two blocks a single chip image sensor and RAM. The sensor chip combines an area image sensor, A/D conversion and control circuitry/signal processing.

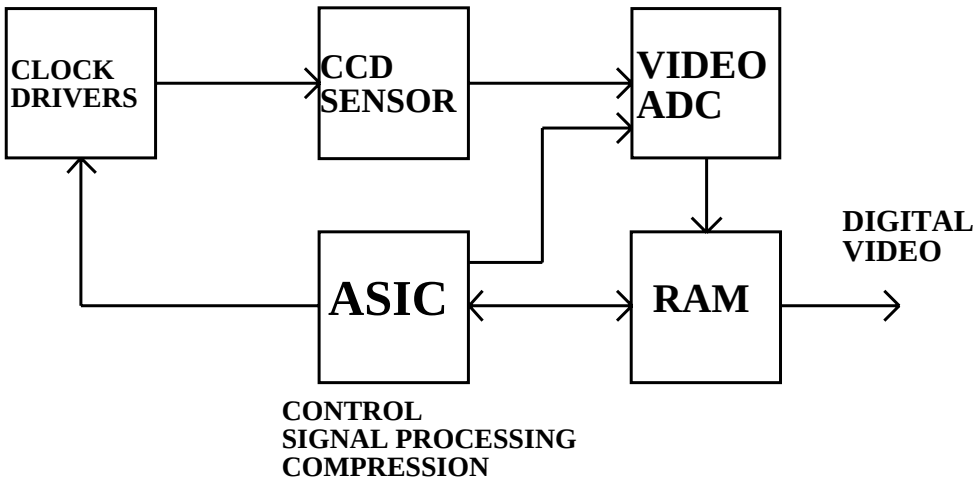


Figure 1.1: Curerent Digital Image Sensor System

Integrating an A/D converter with an image sensor can be done in different ways. One obvious way is to integrate an image sensor with a single A/D converter, as shown in Figure 1.3. This configuration integrates a single high speed A/D converter with the image sensor. Another method takes advantage of the parallelism available on chip, and integrates an image sensor with a semi-parallel A/D converter, as shown in Figure 1.4. This configuration integrates a linear array of A/D converters with the image sensor. Although, semi-parallel configurations typically use one A/D converter per column this classification encompasses any sensor that uses more than one A/D converter located at the edge of the sensor. The method we will discuss in this thesis is an image sensor with a parallel A/D converter, as shown in Figure 1.5. This

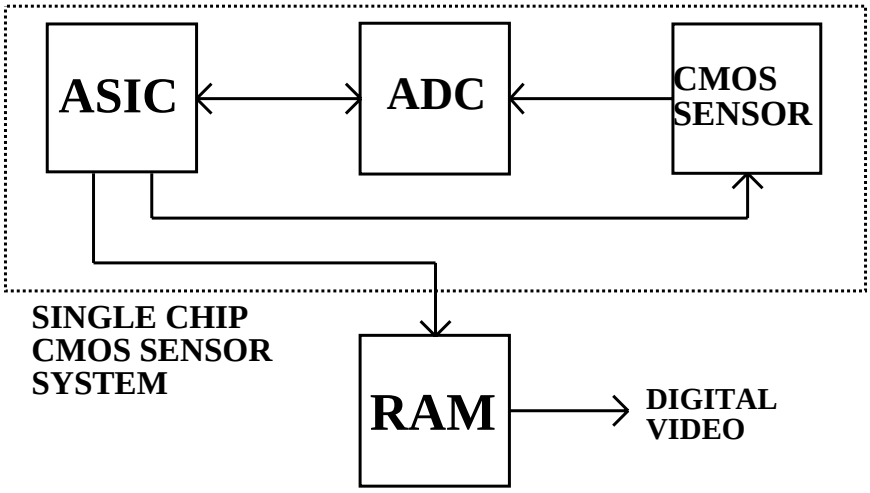


Figure 1.2: Future Digital Image Sensor System

configuration integrates a two dimensional array of A/D converters within the image sensor.

Each of the three configurations described above presents advantages and limitations. The single A/D converter configuration offers design advantages over the other configurations, because the area image sensor and the A/D converter need not be pitch matched and the A/D converters size is not severely limited. This allows the use of standard CMOS design and layout techniques. On the other hand, single A/D converter configurations have several limitations including high power dissipation (caused by the high speed A/D converter), analog communication between the sensor and the A/D converter, and poor process scaling (see Appendix A). The main advantage of a semi-parallel A/D converter configuration is that simple low speed/power A/D converters can be used. Although this is an attractive feature, this configuration suffers from several limitations including complicated layout (because the image sensor and the A/D converters must be pitch matched), analog communication between the sensor and the A/D converters, and poor process scaling (see Appendix A). Finally, the main advantage of a parallel A/D converter configuration

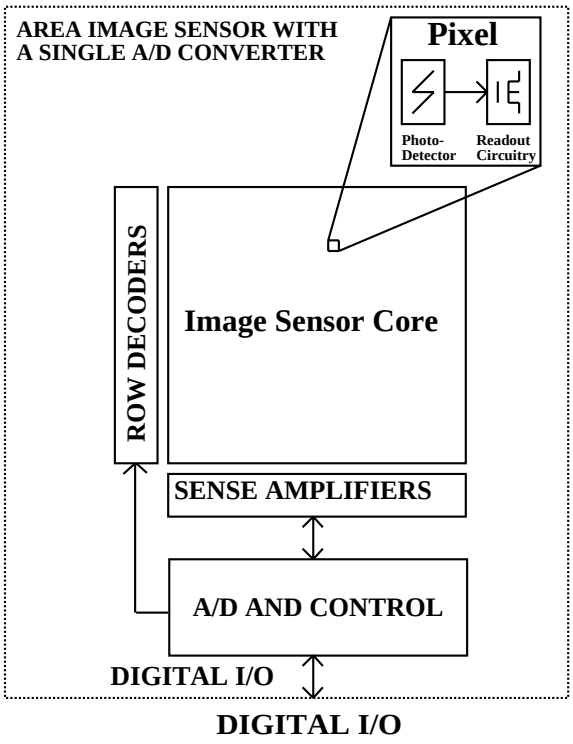


Figure 1.3: Area Image Sensor with a Single ADC

is that very low speed/power A/D converters can be used, and all of the communication between the sensor core and periphery is digital. Parallel A/D configurations also have several limitations including very complicated layout, and severely limited size for the A/D converters. However, modern CMOS processes, such as 0.8μm with three layers of metal, have made it possible to construct an image sensor with a parallel A/D converter. This thesis describes such a construction.

1.1 Previous Work

Several researchers have investigated and built CMOS area image sensors with integrated A/D conversion [54, 56, 55, 24, 3, 32, 46, 18]. Their work shows a progression in the state of the art from single A/D converter configurations to semi-parallel configurations. There has also been a progression in CMOS sensor pixels from scanned photodiodes to active pixels each containing several MOS transistors.

In 1991 Peter Denyer's group at the University of Edinburgh presented the first

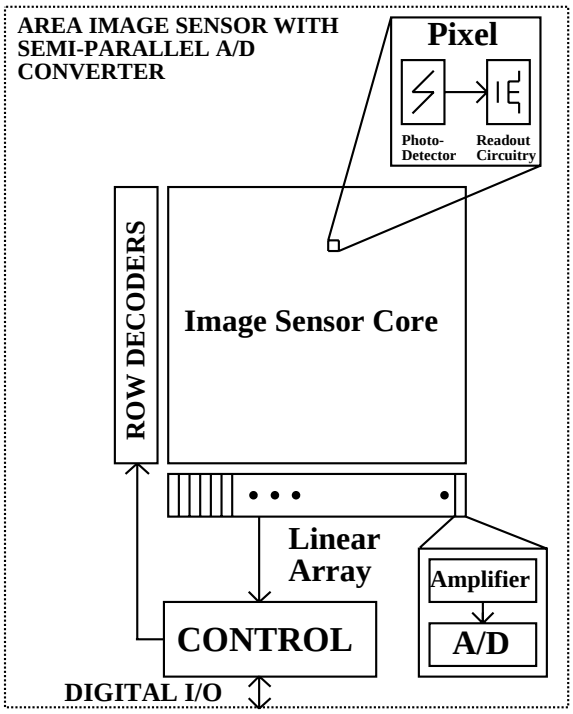


Figure 1.4: Area Image Sensor with semi-parallel ADC

known CMOS area image sensor with a single integrated A/D converter [56]. In [56] they describe a video sensor for finger print capture and verification. A schematic representation of this sensor is shown in Figure 1.6. The sensor consists of a 258×258 pixel array, a unit for image preprocessing and quantization, a 64-cell 2000 Mops/s correlator array, a 16k bit RAM cache, and a 16k bit ROM look up table, LUT. Together with a 64k bit off-chip RAM and a 8051 micro-controller this system can capture and compare a fingerprint against a stored reference print within one second. The on chip quantization was performed using a single high speed successive approximation A/D converter. The sensor power dissipation is 100mW with a 5 volt power supply.

Robert Forchheimer's group at the University of Linkoping and Integrated Vision Products, IVP, developed the first known CMOS area image sensor with semi-parallel A/D conversion [24]. A schematic representation of this sensor, MAPP2200, is shown in Figure 1.8. This sensor contains a 256 × 256 array of photo diodes capable of capturing a full image. The chip includes semi-parallel single slope A/D conversion

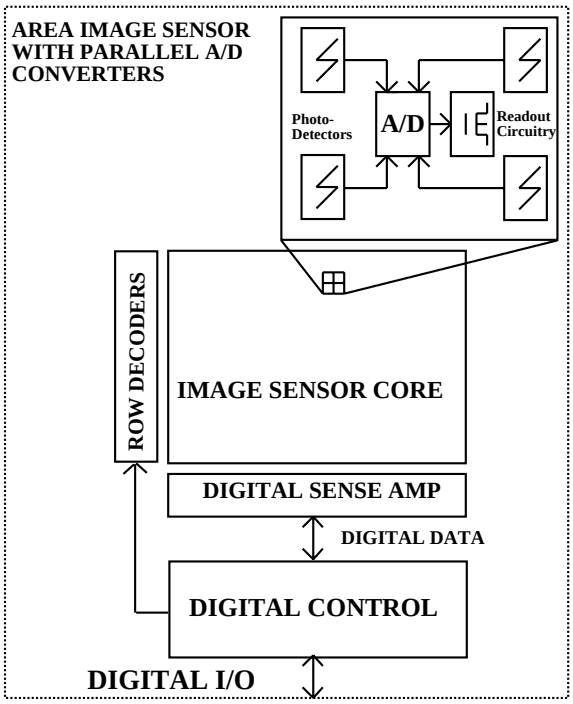


Figure 1.5: Area Image Sensor with parallel ADC

and digital image processing circuitry. The semi-parallel single slope A/D conversion circuitry is depicted in Figure 1.7. The A/D converters maximum precision is 8 bits and the minimum conversion time is 25.6μs. The processor is a line parallel SIMD machine, i.e. there is one processor for each column of the image sensor array. It can process data at a rate of 4MHz/row. The processor can perform common early vision tasks such as filtering, edge detection, histogramming, and correlation at rates of 10-100 frames per second. Simpler tasks such as binary template matching can be performed at more than 1000 frames per second. The MAPP2200 is intended for industrial machine vision applications such as robot navigation, and remove surveillance.

M. Gottardi's group at the Istituto per la Ricerca Scentifica e Technologica developed the POLIFEMO semi-parallel A/D converter image sensor [32]. This sensor contains 128 $\times$ 128 storage mode photo diodes. The chip includes an array of 128 single slope A/D converters and a digital processor to control on chip functionally and communicate with an external microprocessor. The semi-parallel single slope

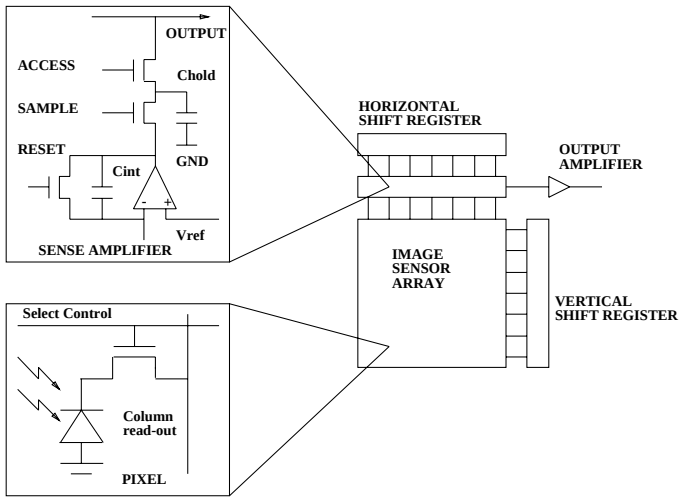


Figure 1.6: Area Image Sensor for Finger Print Recognition

A/D conversion circuitry is identical to the MAP2200. The chip has a maximum operating frequency of approximately 30 frames per second, with 8 bit A/D conversion accuracy. The sensors power dissipation is 125mW with a 5 volt power supply.

Dickinson, Mendis, and Fossum from AT&T and JPL have also developed an image sensor with a semi-parallel A/D converter [18]. This sensor is intended for consumer multimedia applications where low cost and low power video rate image sensors are required. It contains 176×144 photogate active pixels [19]. Each active pixel contains four transistors used for amplification and sensing. Active pixels achieve the lowest reported input referred noise < 30 electrons per frame, of any CMOS image sensor. It uses 176 parallel single slope A/D converters and achieves 8 bits of resolution at 30 frames per second. The A/D converter circuitry is again identical to the MAPP2200. The measured power dissipation of the sensor was 35mW with a 3.5v power supply. Note that the power dissipation is measured without the external D/A converter used in the single slope A/D.

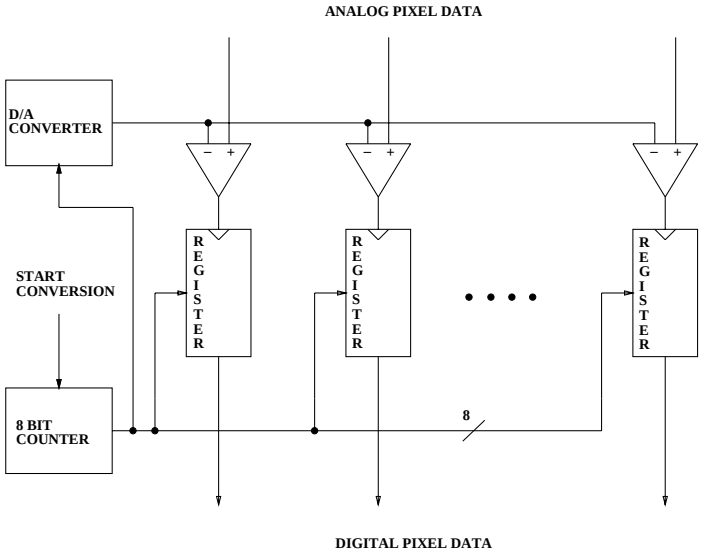


Figure 1.7: Row Parallel Single Slope A/D Conversion

1.2 Summary of This Work

This thesis presents the first known work on CMOS area image sensors with parallel or pixel-level A/D conversion. It is organized into ten chapters. Chapter 2 introduces sigma delta modulation, the A/D conversion technique used by our sensor, and discusses the operation and analysis of one bit first order sigma delta modulators. Chapter 3 introduces the characteristics of the phototransducers used by our image sensor, i.e. CMOS photodiodes and phototransistors. This chapter also compares the devices and discusses their limitations for area image sensors.

Chapter 4 presents a system level description of our CMOS area image sensor with pixel-level A/D conversion. Chapter 5 presents our first generation pixel block circuit, i.e. the basic building block of the sensor. The pixel block circuit is analyzed and results from a 64×64 pixel area image sensor are presented. This chip was fabricated using a 1.2μm two layer metal single layer poly n-well CMOS process. Each pixel block occupies $60\mu\text{m} \times 60\mu\text{m}$ and consists of a phototransistor and 22 MOS transistors.

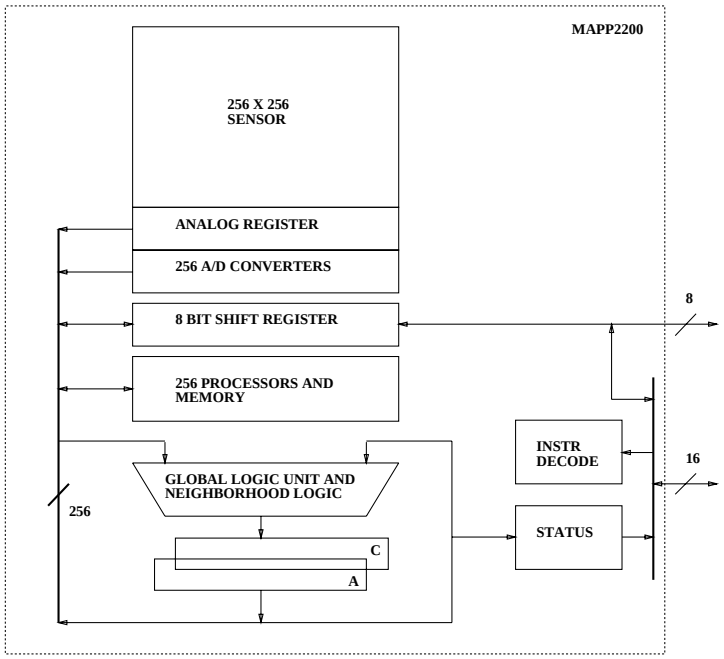


Figure 1.8: MAPP2200

Chapter 6 presents our second generation pixel block circuit. Again, the pixel block circuit is analyzed and results from a 4×4 pixel area image sensor are presented. The chip was fabricated using a 0.8μm three layer metal single layer ploy n-well CMOS process. Each pixel block occupies $30\mu\text{m} \times 30\mu\text{m}$ and consists of a photodiode and 19 MOS transistors.

Chapters 7 and 8 present decimation techniques for sigma delta modulated image data. This includes linear and nonlinear decimation techniques for both still and video image data. Simulation results show the relative trade-off between average SNR and computational complexity. Using these results, the design for an on chip video decimation filter is discussed.

Chapter 9 presents a set of lossless compression algorithms for binary image data. These algorithms are tested on both sigma delta modulated images and standard CCITT FAX images.

Chapter 10 contains concluding remarks and presents several ideas for future work.

Chapter 2

Sigma Delta Modulation

2.1 Introduction

Delta modulation, the first form of oversampled A/D conversion, was first introduced in 1946 [62]. It was intended for quantizing telephone signals using only single bit code words. In 1960 Cutler [16] introduced a quantizer with noise shaping. His idea was to take a measure of the quantization error in one sample and subtract it from the next input sample. This high pass filters the quantization noise. In 1962 Inose, Yasuda, and Murakami combined delta modulation and noise shaping and produced sigma delta modulation. In this chapter we will explore sigma delta modulation and its application to A/D conversion.

When coupled with a digital decimation filter, sigma delta modulation can be used to perform A/D conversion. A block diagram depicting a sigma delta modulator with a digital decimation filter is shown in Figure 2.1. The sigma delta modulator oversamples the analog input at a rate much higher that the Nyquist rate, but each sample only has a few bits of amplitude resolution. The high frequency data from the modulator is then low pass filtered, in order to remove quantization noise, and decimated to the Nyquist rate. We shall show that sigma delta modulation uses noise shaping and oversampling in such a way that resolution in time can be traded for resolution in amplitude.

Oversampling A/D converters have recently become popular because they avoid

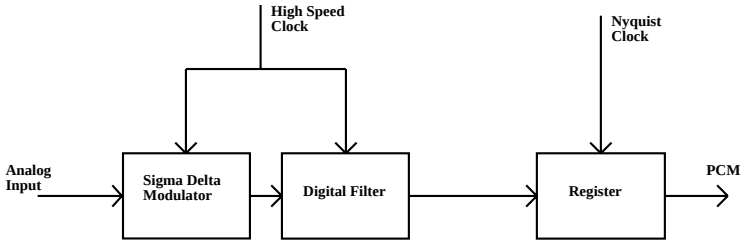


Figure 2.1: Sigma Delta A/D with Decimation Filter

many of the difficulties encounter in traditional Nyquist A/D converters. For example, traditional A/D converters require continuous time analog input filters, high precision analog components, and a low noise substrate environment. Unfortunately, standard CMOS processes make these requirements difficult to implement. The virtue of traditional A/D converters is their use of a relatively low sampling frequency, i.e. the Nyquist rate of the signal, and they directly produce binary weighted data. On the other hand, oversampled A/D converters can use very simple and relatively low precision analog components, but require high speed and complex digital signal processing. This allows us to take advantage of the fact that standard CMOS processes are better suited for providing fast digital circuits than for providing precise analog circuits. Since their sampling rate usually needs to be several orders of magnitude greater that the Nyquist rate, oversampling methods are best suited for relatively low frequency signals. This makes oversampling methods attractive for image sensors, where the Nyquist signal rate is typically around 30Hz.

This chapter is organized into three sections. In Section 2.2 we introduce the basic concepts of scalar quantization. This is done in order to familiarize the reader with notation and concepts that are necessary for analyzing sigma delta modulators. Then in Section 2.3 we present the structure of a one-bit first-order sigma delta modulator and analyze some of its important features. We also discuss the limitations of our

analysis. Section 2.4 summarizes the chapter.

2.2 Scalar Quantization

Sigma delta modulators depend on scalar amplitude quantization. Scalar amplitude quantization is a method of mapping the real line onto a finite set of values. This set of values is called a quantization code book. The mapping between the real line and the code book value is typically done by selecting the code word that minimizes the L1-norm, i.e. the absolute distance between the real values and the code words. As an example, consider a digital thermometer that can measure temperature in degrees Centigrade and temperatures in the range -0.49 to 100.49 it can display the value rounded to the nearest integer. The input value is analog and the output, a two digit number from a set of size 100, is digital.

Quantizers may be uniform or non-uniform quantizers. A uniform quantizer has equal intervals between each quantized value. An example of a uniform quantizer is the digital thermometer. A non-uniform quantizer has unequal intervals between each quantized value.

Quantization adds error to the original signal. We will define quantization error e as

$$e = q(x) - x. \tag{2.1}$$

Where $q(x)$ is the quantized value of the real number x. We will measure this distortion, or quantization noise, using a L2-norm defined by the following equation.

$$\text{distortion} = |x - q(x)|^2 \tag{2.2}$$

We are measuring the distortion caused by quantization based on the L2-norm, therefore we should selected the quantization code words based on minimizing the L2-norm instead of the L1-norm.

Another way of looking at quantization error is by using a stochastic systems approach. This is done by assuming that x, the real value to be quantized, is a discrete time zero mean wide sense stationary (WSS) random process, X. If X_n is

also a white bandlimited process then the quantization error e will also be a zero mean WSS white process, E_n. This leads to the well known linearized quantizer approximation. This approximation allows us to mathematically describe a quantizer as a summer that adds X_n to an uncorrelated white noise source E_n and produces a set of quantized values Y_n. A schematic representation of this process is shown in Figure 2.2. This approximation was shown by Bennett [5] to be valid under the following conditions:

- the quantizer does not overload [28],
- the quantizer has a large number of levels,
- the spacing between code words is small, and
- the probability distribution of pairs of input samples is given by a smooth probability density function [34].

Assuming that the probability distribution of E_n is uniform between $\pm\frac{\Delta}{2}$ the noise power of a uniform quantizer is

$$\sigma_E^2 = \frac{1}{\Delta}\int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} e^2 de = \frac{\Delta^2}{12}. \tag{2.3}$$

Since the quantization noise, E_n, is a sampled random process all of its spectral energy is concentrated at frequencies between $-\frac{f_s}{2} < f < \frac{f_s}{2}$. Where f_s is the sampling frequency and $\tau = \frac{1}{f_s}$ is the sampling period. Since E_n is a white process and all of its power is concentrated between $-\frac{f_s}{2} < f < \frac{f_s}{2}$, the power spectral density of E_n [5] is given by

$$E(f) = \sigma_E^2 \tau. \tag{2.4}$$

The signal to noise ratio of a quantizer is defined as

$$\text{SNR} = 10\log_{10}(\frac{\sigma_s^2}{\sigma_e^2}), \tag{2.5}$$

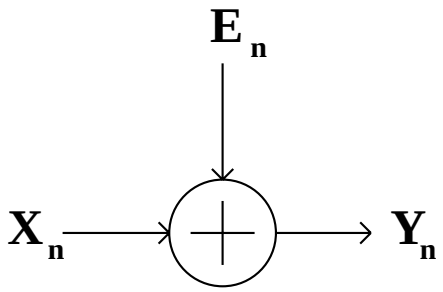


Figure 2.2: Linearized Quantizer Block Diagram

where σ_s^2 is the average signal power defined by

$$\sigma_s^2 = E[X_n^2], \tag{2.6}$$

and σ_n^2 is the average noise power within the signal bandwidth defined by

$$\sigma_n^2 = \int_{-\frac{f_0}{2}}^{\frac{f_0}{2}} E(f)df. \tag{2.7}$$

Where f_0 is the Nyquist frequency of X_n and $E[A]$ is the expected value of the random variable A.

2.3 One-Bit First-Order Sigma Delta Modulation

We will be concerned primarily with *one-bit first-order* sigma delta modulation. More complex modulation techniques are not feasible for pixel-level A/D conversion, due to the limited area available at each pixel. A block diagram of a one bit first order sigma delta modulator is shown in Figure 2.3. *One-bit* refers to the number of quantization

levels used inside the sigma delta modulators feedback loop. *First order* refers to the number of feedback loops, and the order of the high pass quantization noise filtering used in the sigma delta modulator. A first order sigma delta modulator is described

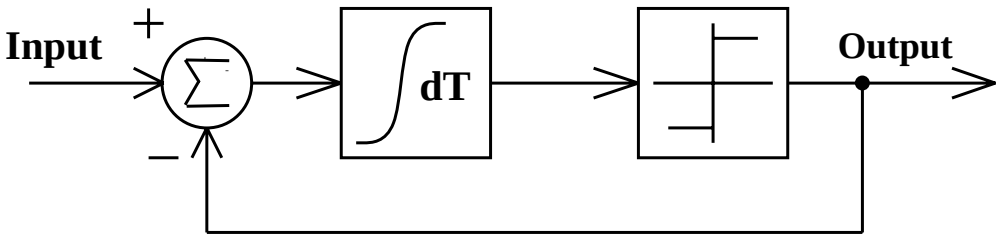


Figure 2.3: First Order One Bit Sigma Delta Modulator

using the following nonlinear difference equation.

$$u_n = u_{n-1} + x_{n-1} - q(u_{n-1}) \tag{2.8}$$

u_n is the state variable of the modulator, x_n is the modulator's input, $q(u_n)$ is the modulator's output, and $q(x)$ is defined as follows

$$q(x) = \begin{cases} b, & x \geq 0 \\ -b, & x < 0. \end{cases} \tag{2.9}$$

The input to sigma delta modulator x is fed to the uniform quantizer via an integrator, and the quantized output y is fedback and subtracted from the input. If the input is larger than zero the average number of positive output samples will exceed the number of negative output samples. But if the input is less than zero the average number of negative output samples will exceed the number of positive output samples. If the input is a constant then the average value of the output of the modulator will converge to the the analog input. This is proved in the following proposition.

Proposition 1 *If x, the input to a first order one bit sigma delta modulator, is constant during a period of L samples and the modulator does not overload then the time averaged output of the modulator will asymtotically converge to x as $L \to \infty$.*

Proof:
Given the nonlinear difference equation for a one bit first order sigma delta modulator

$$u_{n+1} = u_n + x - q(u_n). \tag{2.10}$$

Move u_{n+1} and $q(u_n)$ to the other size of the equality and sum over the first L samples.

$$\sum_{i=0}^{L-1} q(u_i) = \sum_{i=0}^{L-1} (x + u_i - u_{i+1}) \tag{2.11}$$

Now notice that the right side of the equation has a telescoping sum and divide both sides by L.

$$\frac{1}{L} \sum_{i=0}^{L-1} q(u_i) = x + \frac{u_L - u_0}{L} \tag{2.12}$$

There exists an $l = |\frac{2u_l}{\epsilon}|$ such that $L > l$ implies

$$|\sum_{i=0}^{L-1} q(u_i) - x| < \frac{\epsilon}{2}. \tag{2.13}$$

Clearly there also exists l such that $M > l$ implies

$$|\sum_{i=0}^{M-1} q(u_i) - x| < \frac{\epsilon}{2}, \tag{2.14}$$

and so $L, M > l$ implies

$$|\sum_{i=0}^{M-1} q(u_i) - \sum_{i=0}^{L-1} q(u_i)| \leq |\sum_{i=0}^{L-1} q(u_i) - x| + |\sum_{i=0}^{M-1} q(u_i) - x| < \epsilon. \tag{2.15}$$

Hence $\sum_{i=0}^{L-1} q(u_i)$ is a Cauchy sequence and therefore

$$\lim_{L\to\infty} \frac{1}{L} \sum_{i=0}^{L-1} q(u_n) = \lim_{L\to\infty} x + \frac{u_L - u_0}{L} \to x \quad \square \tag{2.16}$$

If we assume that the input signal, x, can be modeled by a zero mean bandlimited WSS random process X_n then we can use the linearized quantizer approximation to analyze the noise performance of this modulator. We will assume that the power spectral density of X_n is

$$X(f) = \begin{cases} \frac{\sigma_x^2}{f_0}, & -\frac{f_0}{2} < f < \frac{f_0}{2} \\ 0, & \text{otherwise.} \end{cases} \tag{2.17}$$

The modulator is sampled at a rate $f_s >> f_0$ and $\tau = \frac{1}{f_s}$ is the sampling period. Figure 2.4 shows the modulator's equivalent circuit using the linearized quantizer approximation. The output of the integrator is

$$U_n = X_{n-1} - E_{n-1}, \tag{2.18}$$

and the quantized output signal is

$$Y_n = q(U_n) = X_{n-1} + (E_n - E_{n-1}). \tag{2.19}$$

This shows how the modulator differentiates the quantization noise, in effect high pass filtering it, while leaving the signal unchanged, except for a unit delay. The power spectral density of the modulation noise

$$N_n = E_n - E_{n-1} \tag{2.20}$$

may be expressed as

$$N(f) = E(f)|1 - e^{-j2\pi\tau}|^2 = 4\sigma_E^2 \tau \sin(\frac{2\pi f\tau}{2}) \tag{2.21}$$

The noise power in the signal band is

$$\sigma_N^2 = \int_{-\frac{f_0}{2}}^{\frac{f_0}{2}} N(f)df \approx \sigma_E^2 \frac{\pi^2}{3}(2f_0\tau)^3. \tag{2.22}$$

Therefore, each doubling of the oversampling ratio $f_0\tau$ reduces this noise by 9dB and provides 1.5 bits of additional amplitude resolution. This improvement in the amplitude resolution requires that the modulated signal be decimated to the Nyquist rate with a low pass digital filter. Otherwise, high frequency noise components will lower the resolution when it is decimated. Note that in [34] it is shown that the assumptions necessary for the linearized quantizer approximation are violated in this circuit. Although this is true it is generally believed that the linearized quantizer analysis provides useful system level insight and produces relatively accurate results [12].

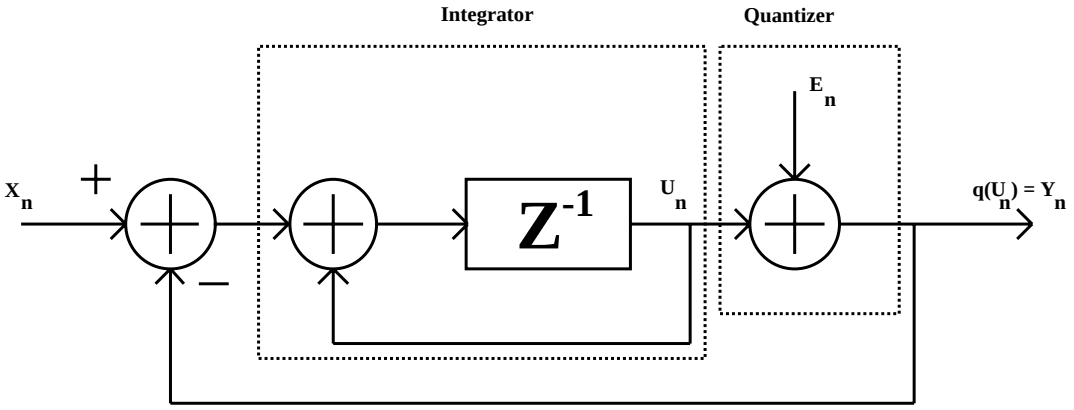


Figure 2.4: Linearize One Bit First Order Sigma Delta Modulator

The above linearized analysis gave us some insights into the circuits operation, but it hides the fact that quantization is inherently a nonlinear process. This is a problem for audio applications, because the nonlinear effects cause dead zones [12] and baseband noise tones [34]. However, software simulation shows that image sensor

applications do not suffer from these problems. It is likely that the eye is less sensitive to these distortions than the ear. In Chapter 7 we will analyze sigma delta modulators as nonlinear systems in order to better understand their operation.

One bit first order sigma delta modulators are robust to analog circuit variations. In order to demonstrate this we will investigate how finite DC integrator gain, one bit quantizer offset errors, and input referred white noise affect the signal to noise ratio. If the integrator has a finite DC gain then its transfer function is

$$H(z) = \frac{z^{-1}}{1 - \alpha z^{-1}}, \tag{2.23}$$

and its DC gain is

$$H(1) = \frac{1}{1 - \alpha}. \tag{2.24}$$

In a lossless integrator α equals one, but in our case it is assumed to be positive and strictly less than one. Using the finite DC gain integrator, the output of the modulator can be expressed as

$$Y(z) = \frac{z^{-1}X(z)}{1 + (1 - \alpha)z^{-1}} + \frac{(1 - \alpha z^{-1})E(z)}{1 + (1 - \alpha)z^{-1}}. \tag{2.25}$$

There is an increase in the low frequency noise and the signal amplitude is decreased by $\frac{1}{2-\alpha}$. If the DC gain of the integrator is at least equal to the oversampling ratio then the baseband noise is only increased by 0.3dB [12]. Therefore, the gain required for the integrator can be quite low. In fact for our pixel-level A/D application we can tolerate integrator gains lower than 100. If the one bit quantizer's threshold is shifted from zero then the modulator output is unchanged, except for a transient period when the integrator is reset. This is true because the one bit quantizer offset is reduced by the gain of the feedback loop. If the one bit quantizer's output levels are shifted and scaled this adds a DC offset term and a gain error to the sigma delta modulator output data. The output offset and gain errors are linearly proportional to the quantizer output errors. In most CMOS image sensors the offset and gain error from pixel to pixel is around 1%. Therefore, our pixel-level A/D converter will require a one bit quantizer with a pixel to pixel offset and gain accuracy of 1%.

Achievable SNR in a sigma delta modulator is constrained by available signal swing at one extreme and noise sources at the other. Noise can arise from power supply or substrate coupling, clock-signal feedthrough, and from thermal and $\frac{1}{f}$ noise generated in the MOS devices. Noise generated inside the modulators feedback loop is suppressed by the feedback action. But noise generated at the input of the modulator adds to the signal and lowers the SNR. Using the linearized model we will analyze the effect of input referred noise assuming all of the input referred noise spectrum $N_{input}(f)$ is white, and

$$P_{innoise} = \int_{-\frac{f_s}{2}}^{\frac{f_s}{2}} N_{input}(f)df. \tag{2.26}$$

The output noise caused by $N_{input}(f)$, after decimation filtering, is

$$P_{outnoise} = \frac{f_0}{f_s} P_{innoise}. \tag{2.27}$$

Therefore, the maximum achievable SNR is given by

$$SNR_{max} = 10\log_{10}(\frac{E[X_n^2]}{P_{outnoise}}) = 10\log_{10}(\frac{E[X_n^2]}{P_{innoise}}) + 3\log_{10}(L). \tag{2.28}$$

Where $L = \frac{f_s}{f_0}$.

One bit first order sigma delta modulators use feedback to reduce quantization noise and improve device sensitivity, but feedback also implies that the loop might be unstable under certain conditions. A sigma delta modulator is defined to be stable if the internal state variable u_n is always bounded above and below by two finite constants. Unlike many feedback control systems, the output of a sigma delta modulator is intended to oscillate. In fact if the modulator becomes unstable the output will stop oscillating. The following proposition proves that one bit first order sigma delta modulators are stable if the input is properly bounded.

Proposition 2 *If $u_0 = 0$ and $x_n = x$ where $-b < x < b \ \forall \ n$ then the modulator will not overload, i.e. $x - b \leq u_n \leq x + b \ \forall \ n$.*

Proof

For $n = 1$

$$u_1 = u_0 + x_0 - q(u_0) \tag{2.29}$$

Using $u_0 = 0$ and the bound on x_n this implies that

$$u_1 = x - b \tag{2.30}$$

$$\Rightarrow x - b \leq u_1 \leq x + b \tag{2.31}$$

Now assume that

$$-b + x \leq u_n \leq x + b. \tag{2.32}$$

Therefore given

$$u_{n+1} = u_n + x_n - q(u_n), \tag{2.33}$$

there are two cases 1) if $u_n \geq 0$.

$$u_{n+1} = u_n + x - b \geq x - b \tag{2.34}$$

$$u_{n+1} = u_n + x - b \leq 2x < x + b \tag{2.35}$$

and 2) if $u_n < 0$.

$$u_{n+1} = u_n + x + b \leq x + b \tag{2.36}$$

$$u_{n+1} = u_n + x + b \geq 2x > x - b \tag{2.37}$$

Now combine the inequalities.

$$x - b \leq u_{n+1} \leq x + b \tag{2.38}$$

Therefore by induction u_n is bounded between $x - b$ and $x + b \; \forall \; n$. $\square$

2.4 Summary

We have investigated one bit first order sigma delta modulators using a linearized quantizer approximation. We determined that oversampling and noise shaping reduce in band noise at a rate of 9dB per octave. We also showed that these modulators are insensitive to circuit variations, making them relatively simple to build in a digital CMOS process. The modulator was also determined to be stable over a wide range of input values.

Chapter 3

CMOS Phototransducers

3.1 Introduction

Each pixel in an image sensor contains a phototransducer. This device converts light into an electrical current. In a standard CMOS process there are three commonly used phototransducers available, p-n junction photodiodes [20, 21, 72, 3, 17, 65, 41, 23, 24, 55, 56, 54, 68, 32], vertical bipolar junction phototransistors [44, 63, 64], and photogates [19, 18]. In this chapter we discuss photodiodes and phototransistors. In an nwell CMOS process, there are three different types of photodiodes and one vertical phototransistor available. The three different photodiodes are created using the nwell-psubstrate junction, the nwell-pdiffusion junction, and the psubstrate-ndiffusion junction. A vertical bipolar junction phototransistor can be created by using the parasitic nwell transistor. Specifically, a pnp phototransistor can be created by diffusing a p+ drain-source region into the nwell. The p+ drain-source diffusion is the emitter the nwell is the base and the psubstrate is the collector of the transistor. Both pwell and nwell processes have the same types of phototransducers except the polarities of the devices are inverted. Figures 3.1 and 3.2 show the cross sections of the photodiodes and the vertical phototransistor in a typical nwell CMOS process.

This chapter is organized as follows: sections 3.2 and 3.3 describe the operation and limitations of photodiodes and phototransistors, respectively. Section 3.4 compares the transducers and comments on their application in image sensors. The final

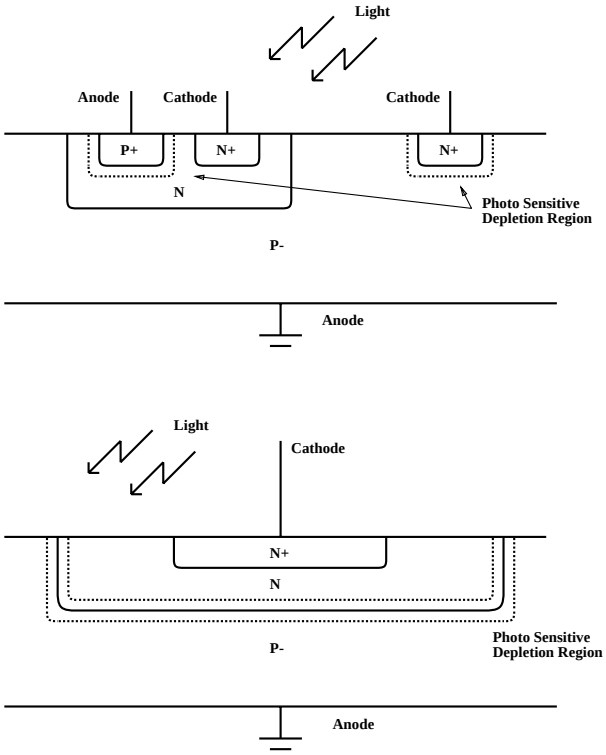


Figure 3.1: Photodiode cross sections in a typical nwell CMOS process.

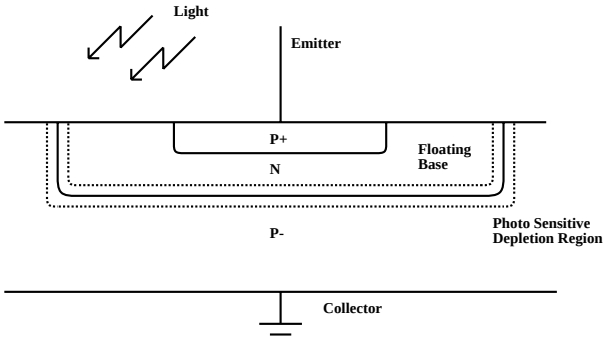


Figure 3.2: Vertical phototransistor cross section in a typical nwell CMOS process.

section presents concluding remarks.

3.2 Photodiodes

3.2.1 Operation

Photodiodes use the photoelectric effect [71] to convert photons into electron-hole pairs. The covalent bonds holding the electrons at atomic sites in the lattice can be broken by incident radiation if the photon energy is greater than the silicon band gap energy. The band gap energy of silicon is 1.124eV which implies that photons with wave lengths less than 1.1μm can theoretically excite carriers from the valence band into the conduction band, i.e. cause photogeneration. Electron-hole pairs that are freed by photogeneration are able to move freely in the lattice and therefore produce currents. Although photogeneration can be used to convert incident radiation into electron-hole pairs these carriers must be efficiently collected if complete recombination is to be avoided. Typical carrier life times in a standard digital CMOS process are on the order of 0.1-10μs. Therefore, the photogenerated electron-hole pairs must be quickly separated and collected.

A simple method of separating and the collecting photogenerated carriers is accomplished by using a reverse biased p-n junction, i.e. a photodiode. In order for a reverse biased p-n junction to maintain electrical equilibrium the diffusion of carriers across the junction and the reverse bias voltage must be balanced by a strong electric field in the depleted quasi-neutral region. If photogeneration occurs within the depleted quasi-neutral region, then the built-in electric field will quickly separate and collect the electron-hold pairs, as shown in Figure 3.3. This will form a photo-leakage current through the diode that is proportional to the incident light intensity. This leakage current can be quantified by the following equation [8]

$$I_{photo} = \frac{\eta q P_s}{h\nu} \tag{3.1}$$

where η is the quantum efficiency of the photodiode, P_s is the incident light power,

h is Plancks constant, and ν is the wavelength of the light. Quantum efficiency is defined as

$$\eta = \frac{\text{number of collected electron hold pairs}}{\text{number of incident photons at wavelength } \nu}. \tag{3.2}$$

Carriers that are photogenerated in the bulk n or p regions can either recombine in the bulk or diffuse through the bulk to the depletion region and add to the photocurrent. In a standard CMOS process, carriers photogenerated in the bulk typically do not recombine before they can enter the depletion region. This implies that the light sensitive portion of the photodiode can be larger than the depletion region between the p-n junction.

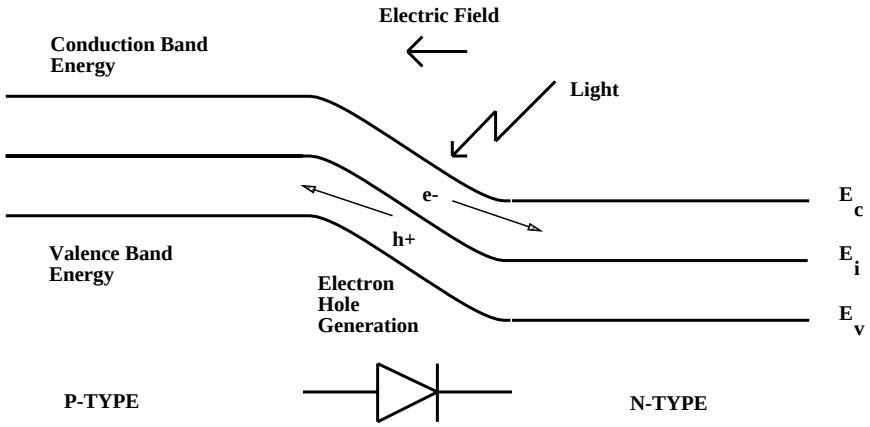


Figure 3.3: Energy profile of photodiode.

3.2.2 Spectral Characteristics

The absorption of light by silicon is wavelength dependent. Short wavelength radiation, i.e. high energy photons, are quickly absorbed at shallow depths, and long

wavelength radiation, i.e. low energy photons, penetrate much deeper before they are absorbed. The absorption length, defined as the distance in which $\frac{1}{e}$ of the incident photons have been absorbed, is 0.3μm for blue light (wavelength 475nm), and 3μm for red light (wavelength 650nm) [17]. This behavior is due to the probability distribution of photons with a given energy exciting a carrier from the valence band into the conduction band. High energy photons have a high probability of exciting a carrier from the valence band into the conduction band, and low energy photons have a low probability of exciting a carrier from the valence band into the conduction band. A simple derivation of absorption length can be preformed by assuming that a photon of energy $h\nu$ will be absorbed by an atom with probability $Pr_{h\nu}$. This is just the photoelectric effect. After interacting with N different atoms the probability that a photon has been absorbed is

$$Pr_{absorption} = 1 - (1 - Pr_{h\nu})^N. \tag{3.3}$$

If we assume that the atoms are placed end to end from the surface of the silicon to a depth x. Where $x = Na$, and a is the diameter of an atom. Then we can write

$$Pr_{absorption}(x < X) = 1 - (1 - Pr_{h\nu})^{\frac{x}{a}}. \tag{3.4}$$

Now assuming that X is a continuous variable random variable, we take the first derivative of the above equation and the distribution of X, $f(x)$, is

$$f(x) = \frac{1}{a} \ln(\frac{1}{1 - Pr_{h\nu}}) e^{-\frac{x}{a} \ln(\frac{1}{1-Pr_{h\nu}})}. \tag{3.5}$$

This is an exponential distribution with a mean value of $\frac{a}{\ln(\frac{1}{1-Pr_{h\nu}})}$. Note that mean value of X is equal to the distance in which $\frac{1}{e}$ of the incident photons have been absorbed.

Wavelength dependent absorption means that photodectors formed from pn junctions with different junction depths will have different spectral responses. In [17] Delbruck shows the quantum efficiency verses light wavelength of p-n junctions in a 2μm CMOS process.

3.2.3 Dark Current and Other Noise Sources

What limits the absolute light detection capabilities of photodiodes? Junction leakage current and white noise limit the minimum detectable light intensity. Junction leakage current is a function of temperature and the doping characteristics of the photodiode. Assuming a short base diode model [48], because of the long recombination times in a standard CMOS process, the reverse bias current can be predicted by

$$I_{reverse} = Aqn_i^2(\frac{D_p}{N_d W_B} + \frac{D_n}{N_a W_E})(e^{\frac{qV_a}{kT}} + 1) \tag{3.6}$$

Where A is the cross sectional area of the diode, q is the charge on an electron, n_i is the intrinsic carrier concentration of silicon (around 1.45e10 at 23 degrees centigrade), D_p is the diffusion constant of holes, N_d is the donor concentration in the n silicon, W_B is the distance, in the n silicon, between the space charge region and an ohmic contact. D_n is the diffusion constant of electrons, N_a is the acceptor concentration in the p silicon, W_E is the distance, in the p silicon, between the space charge region and an ohmic contact. V_a is forward bias voltage across the diode, k is Boltzmann's constant, and T is absolute temperature. The diffusion constants decrease as approximately $T^{-1.4}$ [48], but the intrinsic carrier concentration increases as $T^{\frac{3}{2}}e^{\frac{1}{T}}$ [48]. Therefore, $I_{reverse}$ increases with increasing T. By controlling the fabrication process and operating the sensor at low temperatures it is possible to significantly reduce dark current in a standard CMOS process. In fact CCDs used in astronomy are routinely cooled to 77K to reduce the dark current and increase the sensors SNR [69].

White noise in photodiodes is generated by shot noise. This is confirmed by Delbruck in [17]. The white noise floor is given by

$$I_{noise}^2 = 2q(I_{photo} + I_{reverse})BW. \tag{3.7}$$

This is derived in [8]. q is the change on an electron, I_{photo} is the photon induced leakage current of the diode, and BW is the bandwidth of the sensor. The bandwidth of the photodiode is limited by the depletion capacitance. Since the photodiode can be modeled as a capacitor being charge by a current source $BW \sim \frac{I_{photo}}{C_{depletion}V_{dd}}$. Note

that if $I_{reverse} \approx 0$, I^2_{noise} is still non-zero. This is the photon noise limit of the photodector.

Another form of noise is fixed pattern noise. Fixed pattern noise is defined as the photo-response variation between adjacent photodiodes. This variation is dominated by the area of the diodes Therefore, in a standard CMOS process the fixed pattern noise is controlled by the relative size of the photodiode to the minimum lithographic feature size. The magnitude of this noise is usually less than 1% for a photodiode in a standard CMOS process.

3.3 Phototransistors

Phototransistors also use the photoelectric effect to convert photons into electron-hole pairs. The electron-hole pairs generated by the photoelectric effect are separated and collected using a reversed biased p-n junction in very close proximity to a forward biased p-n junction. As with a photodiode, incident radiation in the reversed biased p-n junction, i.e. the collector-base junction, causes a photocurrent that is proportional to the light intensity. A typical potential profile of a pnp phototransistor is shown in Figure 3.4 As the photocurrent enters the base region of the bipolar transistor it is effectively multiplied by β, the transistor current gain. Since β is typically much greater than one the phototransistor can have a quantum efficiency greater than one in the visible spectrum.

3.3.1 Spectral Characteristics

The spectral characteristics of phototransistors are dominated by the same effects described in the previous section, i.e. junction depth. Again in [17] Delbruck shows the quantum efficiency verses light wavelength of a pnp phototransistor in a 2μm CMOS process.

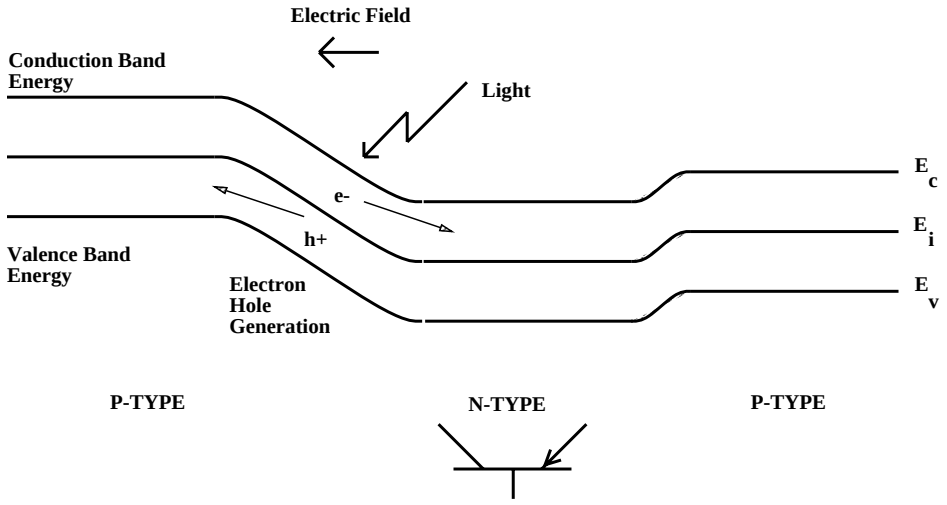


Figure 3.4: Energy profile of pnp phototransistor.

3.3.2 Dark Current and Other Noise Sources

As with photodiodes, phototransistors are detection limited by both dark current and white noise. The dark current in a phototransistor is the result of the reverse bias leakage current in the base-collector junction. This current is multiplied by β, and the resulting emitter current is the total dark current of the device. Since β is typically larger than one the dark current generated by a phototransistor should be larger than that of a photodiode with the same junction area.

White noise in phototransistors is dominated by shot noise in each diode junction. In a common collector configuration the white noise seen at the emitter is

$$I_{noise}^2 = 2q(I_{photo} + I_{reverse})(\beta + 1)BW. \tag{3.8}$$

Where $I_{reverse}$ is the leakage current of the base-collector junction. The bandwidth of the phototransistor is determined by the base-collector depletion capacitance $C_{depletion}$ and the photocurrent. This assumes that the capacitance seen between the emitter

and AC ground is less than $(\beta + 1)C_{depletion}$. Therefore, the white noise in a phototransistor is $(\beta + 1)$ times larger than that of a photodiode with the same junction area.

Although phototransistors can have quantum efficiencies greater than one, this can be a drawback. Specifically, the measured β of an array of phototransistors can vary by more than 20% over a chip, assuming an emitter bias current of 1nA-100μA. The β variation is caused by poor control of the base width. In an image sensor application this leads to fixed pattern noise. The β of a phototransistor is also very temperature sensitive, bacause of the diffusion based carrier transport mechanism within the base region.

3.4 Phototransducer Comparison

Photodiodes have lower white noise and dark current per unit area than phototransistors. Photodiodes also have lower fixed pattern noise than phototransistors. This implies that photodiodes are better suited for image sensor applications than photodiodes. The only possible advantage that phototransistors have over photodiodes is higher quantum efficiency. But at low light levels, where the emitter current of the phototransistor is below 100pA, the quantum efficiency of the phototransistor is approximately equal to the quantum efficiency of a photodiode. This occurs because β is a decreasing function of emitter current. Moreover, when recombination within the base becomes a significant proportion of the emitter current β becomes a decreasing function of emitter current [48].

3.5 Summary

We have described the operation and limitations of photodiodes and phototransistors. We have also concluded that photodiodes are better suited to image sensor applications, because of lower fixed pattern and white noise.

Chapter 4

Area Image Sensors with Pixel-Level ADC

4.1 Introduction

Charge-coupled devices (CCD) are at present the most widely used technology for implementing area image sensors. However, CCD image sensors suffer from the following problems:

- low yields[1],
- high power consumption [1],
- sensor size / SNR limitations due to the shifting and detection of analog charge packets [25],
- and data is communicated off chip in analog form.

Several alternatives to CCD area image sensors that use standard CMOS technology have been developed. Self scanned photodiode arrays have been used to produce

[1] A typical CCD image sensor has over 50% of its area covered by either thin or inter-poly oxide. This causes low yeilds, due to oxide punch through, when compared with a CMOS image sensor that has less than 5% of its area covered by thin or inter-poly oxide.

both binary and gray scale image sensors [23, 24, 56]. Bipolar junction phototransistor arrays [64], and charge injection arrays [47] have also been used. However, these alternatives suffer from low resolution due to limited pixel observation time, limited SNR due to analog sensing, and as with CCDs, data is communicated off chip in analog form.

In this chapter we describe an area image sensor that can potentially circumvent the limitations of CCDs and their alternatives. The proposed image sensor uses a standard CMOS process. It therefore benefits from the higher yields obtained by CMOS processes. Digital circuitry for control and signal processing can be integrated with the sensor. Moreover, CMOS technology advances such as scaling and extra layers of metal can be used to improve pixel density and sensor performance.

This sensor is also designed for low power operation, wide dynamic range, and programmable SNR, i.e. each pixel can be represented with a variable number of bits. All of these features are intended to produce low cost area image sensors that can be used in video phones, portable digital cameras, and machine vision applications.

This chapter is organized as follows: Section 4.2 presents a system level overview of our area image sensor with pixel-level A/D conversion. Section 4.3 describes the temporal response differences between our sensor and conventional CMOS or CCD sensor. Finally, Section 4.4 concludes the chapter.

4.2 System Description

A block diagram of our area image sensor with pixel-level A/D conversion is shown in Figure 4.1. The sensor is constructed of a two dimensional array of pixel blocks. Pixel blocks are addressed row by row using an m to 2^m decoder. Each pixel block is connected to a bit line using a pass transistor. The bit lines are "read" using an array of digital sense amplifiers. This topology is exactly the same as read only memory circuits. Each pixel block converts the analog light intensity into a digital code. This is shown in Figure 4.2. Note that during each frame of video the pixel block generates L bits of data. The entire system is synchronous and after each clock pulse every pixel block produces one bit of data. This generates a two dimensional array of bits or a

"bit plane". Figure 4.3 presents a visual representation of the bit planes generated by the sensor during a frame of video. One frame of video data consists of L bit planes.

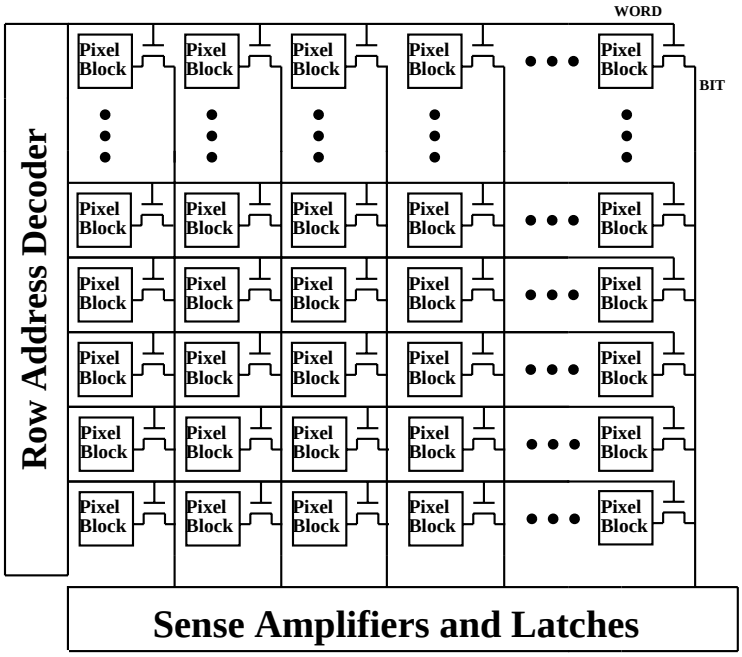


Figure 4.1: Image Sensor Chip Functional Block Diagram

Since we have selected to use pixel-level A/D conversion, each A/D converter must be constructed using a very small amount of silicon area. The A/D converter's size is inversely proportional to the number of pixels in the sensor, and the fill factor of each pixel. Conventional Nyquist A/D conversion techniques (such as flash, successive approximation and single slope) require too much silicon area to be viable for pixel-level A/D conversion. Therefore, we investigated oversampled A/D conversion, and compared various techniques based on reliability, required oversampling ratio for 8 bits of resolution, and size. We found that one bit first order sigma delta modulation provided a reasonable compromise between all of our requirements. Moreover, one bit first order sigma delta modulation can be performed with very little silicon real estate, and is very robust to process variation and system noise.

A block diagram of the pixel block circuit is shown in Figure 4.2. After an image is focused on the chip the sigma delta modulators are reset via the global reset signal.

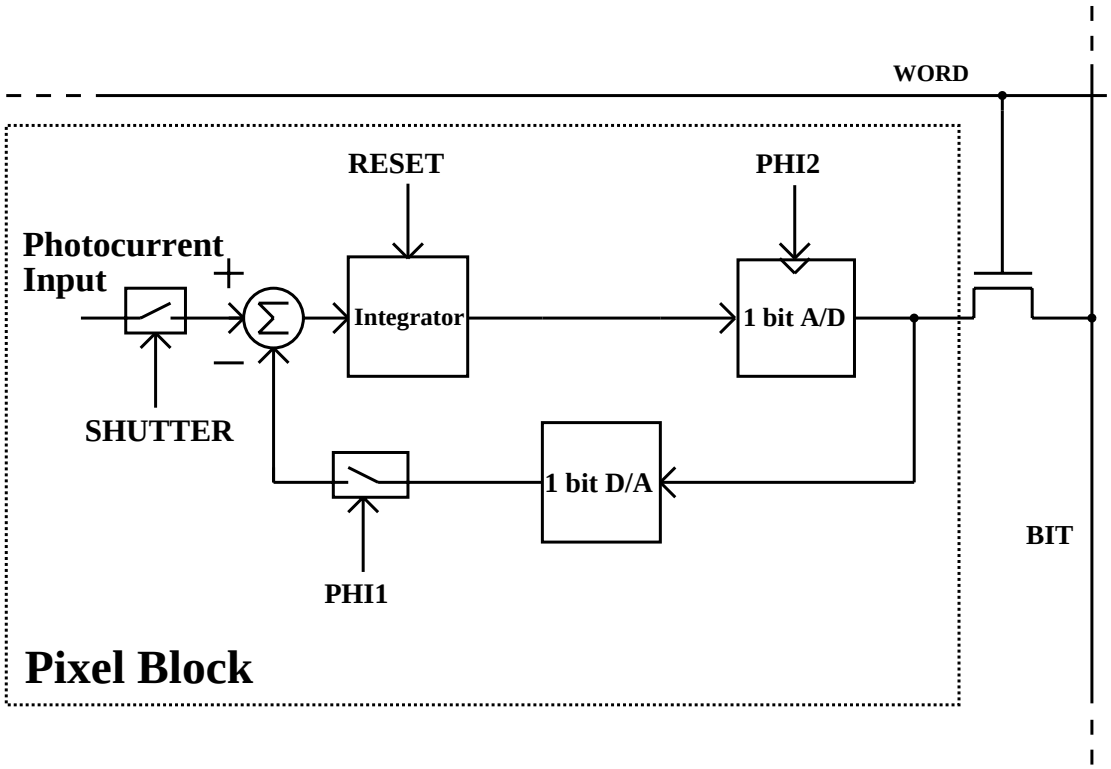


Figure 4.2: Pixel Block

Reset initializes the integrator to a known state. The shutter duty cycle is then globally set to maximize image SNR without exceeding the maximum input to the data conversion circuitry. The shutter circuitry provides an electronic method for reducing the effective light intensity seen by the sensor. This is achieved by attenuating the photocurrent produced by the phototransistor or photodiode. Next the sigma delta modulators are globally clocked at a rate f_s above the image frame rate $2f_d$. f_s is typically much larger than $2f_d$, approximately 32 to 64 times larger. This is necessary since a sigma delta modulator reduces quantization error at the cost of extra data. At the end of each clock cycle the outputs of the sigma delta modulators generate a bit plane. Then each bit plane is read out row by row. A frame is fully captured using a number of bit planes determined by the target SNR. Adjusting the number of bit planes per frame allows digitally programmable SNR. Using the theoretical linear analysis in Chapter 2 the number of bit planes L needed, as a function of SNR is given by

$$L = 2^{\frac{SNR-5.2dB}{9}}. \tag{4.1}$$

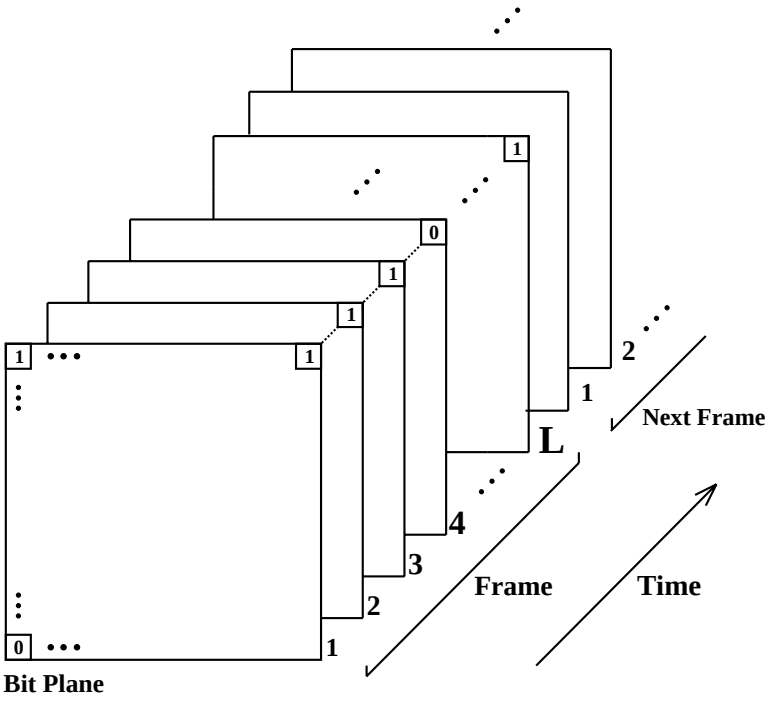


Figure 4.3: Bitplanes

For example, if an SNR of 48dB, 8 bits, is required then L must be greater than 60, and if an SNR of 20dB, 3.3 bits, is required then L must be greater than 7. This implies that a sigma delta modulated pixel will produce more, redundant, data than a standard CMOS or CCD image sensor. This may become a problem if the image sensor is very large.

The digitized pixel values are reconstructed using a decimation filter [11]. Depending on the type of application in which the sensor is used, this reconstruction may be implemented in software, using special purpose hardware external to the sensor, or integrated with the sensor. In a low resolution application where no local reconstruction is needed, e.g. video phone or surveillance camera, the sensor digital output is compressed and immediately transmitted. Reconstruction is done at the receiving end using general or special purpose hardware. This scheme is shown in Figure 4.4. If the image is to be displayed or processed locally, one or more decimation filters are integrated with the sensor and an external RAM is used. The pixel values stored in the RAM are recursively updated using the decimation filters and the new bits from

the corresponding sigma delta modulators. This scheme appears feasible even for a sensor with as many as 1 million pixels operating at 30 frames per second at 8 bits per pixel resolution. Figure 4.5 shows a block diagram of this scheme.

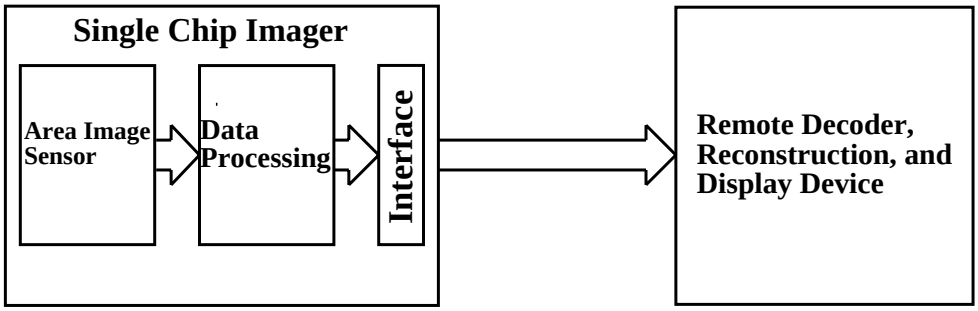


Figure 4.4: Remote Decimation Scheme

Decimation filtering is the process of down sampling the data to the Nyquist data rate, and separating the quantization noise from the signal. As discussed in Chapter 2 the quantization noise in a one bit first order sigma delta modulator is high pass filtered by the system. Using a band limited signal assumption, signals of interest are located between DC and half the Nyquist sample rate. Therefore, if a low pass filter is used during decimation most of the quantization noise can remove without affecting the input signal data. Throughout the rest of this thesis we will consider "decimation" as the process of both decimating and filtering the sigma delta modulated data. None of the sensors described in this thesis have integrated decimation, but Chapters 7 and 8 discusses the trade-offs associated with on chip decimation.

4.3 Temporal System Response

By using oversampling we solve the sinc filtering problem encountered with conventional image sensors [28]. We begin this section by describing the sinc distortion problem and how it applies to image sensors. Then we conclude by showing that our oversampled image sensor does not suffer from sinc distortion.

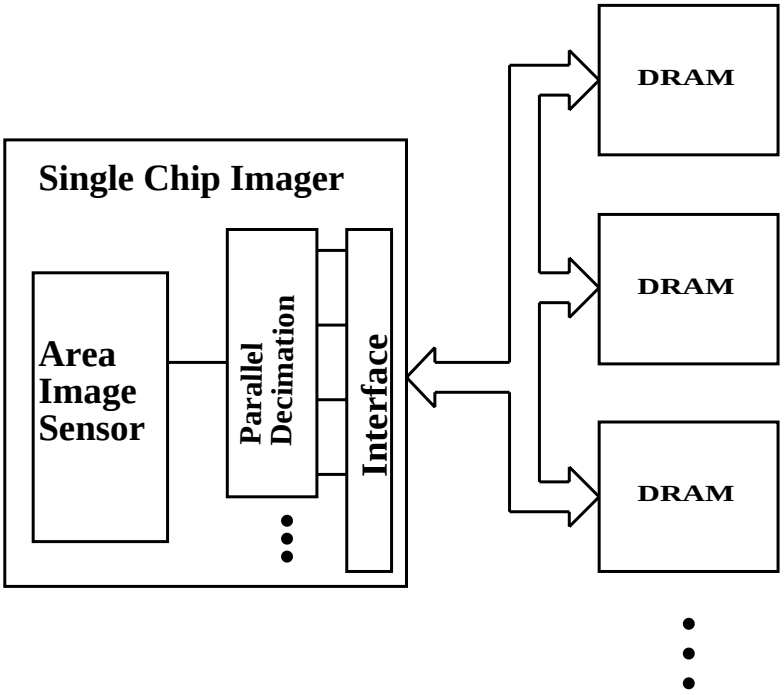


Figure 4.5: Local Decimation Scheme

Most area image sensors integrate photocurrent during each video frame, and then transfer the integrated charge to the edge of the sensor. The integrated charge is then converted into an analog voltage. This is a method of periodically sampling the image data. The sampling, or frame, rate of the sensor is typically determined by either the Nyquist data rate or the frequency response of the viewer. In the former case the original data can be recovered from the sampled data, but because the image sensor performs "integration sampling" and not point sampling high frequency information is attenuated. Therefore, to recover the original signal processing must be used. To clarify this point we will need to develop some notation. Assume that the original video signal at a given pixel is $v(t)$, and define

$$\sqcap(x) = \begin{cases} 1, & \text{if } \frac{-1}{2} < x < \frac{1}{2} \\ 0, & \text{otherwise,} \end{cases} \tag{4.2}$$

$$\delta(x) = \lim_{\tau \to \infty} \int_{-\infty}^{\infty} \tau^{-1} \sqcap (\frac{x}{\tau}) dx, \tag{4.3}$$

and $\mathcal{F}(x(t))$ is the Fourier transform [9] of x(t)

$$\mathcal{F}(x(t)) = \int_{-\infty}^{\infty} x(t)e^{-j2\pi tf}dt. \tag{4.4}$$

The inverse Fourier transform is

$$x(t) = \int_{-\infty}^{\infty} \mathcal{F}(x(t))e^{j2\pi tf}df. \tag{4.5}$$

Now we can describe the integration sampled video signal, $v_{isample}(t)$, as

$$v_{isample}(t) = (v(t) * \frac{1}{T} \sqcap (\frac{t}{T}))\frac{1}{T}\sum_{n=-\infty}^{\infty} \delta(t - nT) \tag{4.6}$$

Where $x(t) * y(t)$ is the convolution of $x(t)$ and $y(t)$, and T is the sampling period. The Fourier transform of $v(t)_{isample}$ is

$$\mathcal{F}(v_{isample}(t)) = (\mathcal{F}(v(t))\frac{\sin(\pi Tf)}{\pi Tf}) * \sum_{n=-\infty}^{\infty} \delta(f - \frac{n}{T}). \tag{4.7}$$

This shows that $\mathcal{F}(v(t))$ is multiplied by $\frac{\sin(\pi Tf)}{\pi Tf}$ in the frequency domain causing the attenuation of high frequencies. If we assume that the power spectral density of the signal is uniform within the Nyquist bandwidth then the mean squared error caused by $\frac{\sin(\pi Tf)}{\pi Tf}$ is

$$MSE = \int_{-\frac{1}{2}}^{\frac{1}{2}} (1 - \frac{\sin(\pi Tf)}{\pi Tf})^2 df. \tag{4.8}$$

Figure 4.6 shows the total mean squared distortion caused by integration sampling versus the oversampling ratio.

In order to illustrate why one-bit first-order sigma delta modulators do not suffer from sinc distortion, we compare the amount of sinc distortion versus quantization noise for a given oversampling ratio. Assuming an oversampling ratio of 32, Figure 4.6 shows that the sinc distortion will be -75dB, but the the quantization noise floor is -40dB. Since sinc distortion falls by 12dB per octave and the quantization noise falls by 9dB per octave, the amount of distortion caused by the quantization noise will always

upper bound the sinc distortion caused by integration sampling. If high frequency distortion is of concern, a CCD image sensor must sample at 7 times the Nyquist rate to achieve 8 bits of resolution compared with 60 times for our image sensor. This assumes that no signal processing is preformed on the sampled data, i.e. frequency compensation. Therefore, if high frequency distortion is of concern a sigma delta modulated pixel produces approximately the same amount of data as a conventional image sensor requiring 8 bits of resolution.

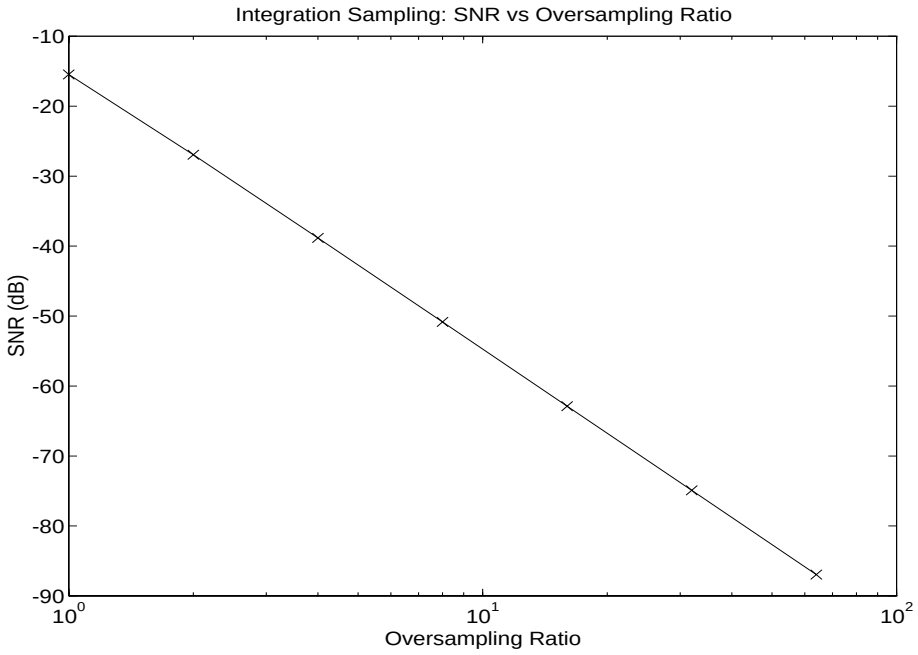


Figure 4.6: Sinc Distortion Caused by Integration Sampling

4.4 Summary

The key features of our CMOS area image sensor with pixel-level A/D conversion are:

- A/D conversion is performed at each pixel using first order sigma delta modulation.

- Region of interest windowing can be performed using the random access structure of the sensor, and
- SNR can programmed by selecting the oversampling ratio of the sigma delta modulators.

Oversampling causes our sensor to generate more data than a standard CMOS or CCD sensor with Nyquist A/D conversion. For a resolution of 8 bits per pixel, our sensor produces approximately seven times as much data as a standard CMOS or CCD sensor. Decimation is used to convert sigma delta modulated data into binary weighted data. An electronic shutter is integrated within the sensor, allowing a wide range of lighting conditions to be imaged. The shutter is programmable and removes the need for an external mechanical shutter within the lens. The sensor also has lower average mean squared error frequency distortion than standard CMOS and CCD sensors using integration sampling. For a resolution of 8 bits per pixel, a frame rate of 30Hz, and average mean squared error frequency distortion of less than -48dB our sensor, before decimation, generates approximately the same amount of data as a standard integration sampled CMOS or CCD sensor with Nyquist A/D conversion.

Chapter 5

First Pixel Block Circuit Design

5.1 Introduction

In Chapter 4 we presented a system level description of our area image sensor with pixel-level A/D conversion. This chapter describes our first implementation of this system. We begin by describing the pixel block circuit. Then the pixel block circuits noise and speed limitations are analyzed. In Section 5.4 test results from a 64×64 pixel block area image sensor are presented [26]. Finally, in Section 5.5 we discuss the limitations of this sensor and compare it with a typical CCD sensor. A die photograph of this chip is shown in Figure 5.1.

5.2 Circuit Description

A circuit schematic of the pixel block is given in Figure 5.2. The circuit consists of three sections, a phototransducer and integrator, a one bit A/D converter, and a one bit D/A converter. The phototransistor **Q1** is a vertical bipolar pnp transistor; the emitter is formed using source-drain p+ diffusion, the base is the n-well surrounding the emitter and the collector is the p- substrate. The n-well is exposed to light, while the rest of the circuitry is covered with metal to reduce the chance of photon induced latch-up. The size of the phototransistor was designed so that under office lighting conditions the emitter current would be 1nA. Control of the input photocurrent is

Figure 5.1: Die Photograph of Original 64×64 Pixel Block Sensor

achieved by setting the duty cycle (the ratio between the on and off times) of the shutter input **SHUTTER** — the higher the duty cycle the larger the input photo current. When the shutter is on, i.e. **SHUTTER** is low, current flows through **M1** and **M2** until the voltage on the base of **Q1** is such that the current through **M2** is equal to the photon induced base current. This effectively turns off any emitter current in **Q1**. When the shutter is off, i.e. **SHUTTER** is high, the phototransistor operates normally. Current from the phototransistor is integrated on **C1**. **C1** is formed using an NMOS transistor. The size of the capacitor, 650fF, was selected such that after 1ms an emitter current of 1nA would cause a 1.5v change on **C1**. Using an NMOS capacitor limits the linear operating region of the integrator between V_{dd} and the threshold voltage of the NMOS device.

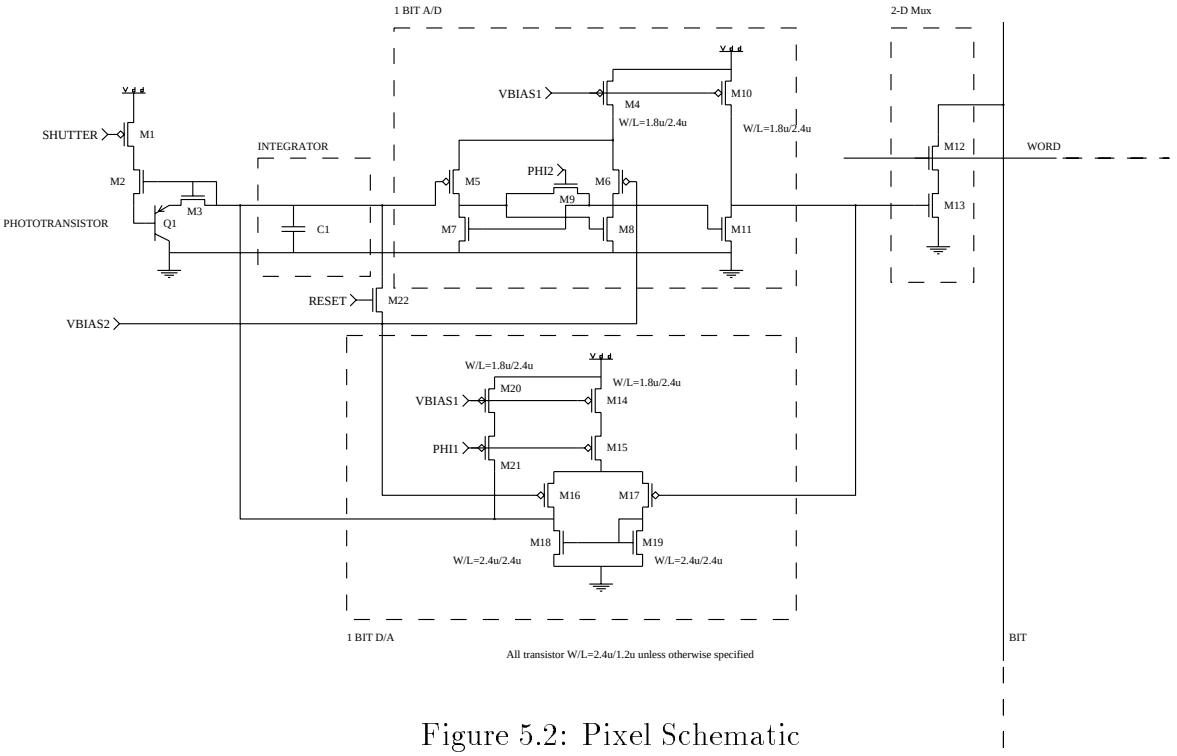


Figure 5.2: Pixel Schematic

The voltage on **C1** is quantized using a regenerative latch, i.e. a one bit A/D converter, clocked via **PHI2**. **M5** and **M6** form a matched differential pair and amplify the difference between the voltage on **C1** and **VBIAS2**. The differential voltage is latched using the cross coupled pair **M7** and **M8**. The latching operation

is performed on the falling edge of **PHI2**. The state of the latch is held while **PHI2** is low and **M9** is off. **M10** and **M11** restore the output of the latch to full swing. The quantized output from the one bit A/D is converted into a current using a one bit D/A converter and fed back to the input capacitor **C1**. The duty cycle of **PHI1** and the voltage **VBIAS1** control the magnitude of the feedback signal current. The input of the D/A converter is the gate of **M17**. For the sake of simplicity, assume that **PHI1** is low and **VBIAS2** = $\frac{V_{dd}}{2}$. If the input is high then **M17** is off and **M16** is on, this causes current from **M14** and **M20** to flow into **C1**. This is the active feed back state. If the input is low then **M17** is on and **M16** is off. Therefore, the current from **M14** flows through **M17** and **M19**, and this current is then mirrored through **M18**. The mirrored current is subtracted from the current generated by **M20**. If the current mirror, **M19** and **M18**, is perfect and **M14** is exactly matched to **M20** then the output current is zero. In a real circuit the current mirror will not be perfect and **M14** and **M20** will not be exactly matched. This will cause offset errors and fixed pattern noise.

The pixel block was designed such that the clocking and readout techniques would closely mimic the techniques used in a standard RAM. **PHI1** and **PHI2** constitute a two phase nonoverlapping clock. At the completion of each two phase clock cycle a single bit is produced. The bit is read by enabling the word line **WORD**. If the bit is high the precharged bit line **BIT** is pulled down and sensed by a simple single-ended sense amplifier. A schematic of the sense amplifier is shown in Figure 5.3. Both the word line selector circuits and the sense amplifier designs are not critical because the pixel blocks were designs to operate at a sampling rate of 1kHz. This corresponds to a maximum switching frequency of 64kHz. Our main concern with the word line selector circuits and sense amplifiers was power dissipation. This can only be reduced by lowering the power supply and or switching frequency.

5.3 Circuit Analysis

All of the MOS circuits used in the pixel block are operated below threshold in order to reduce power dissipation, and increase voltage gain [70]. The characteristics and

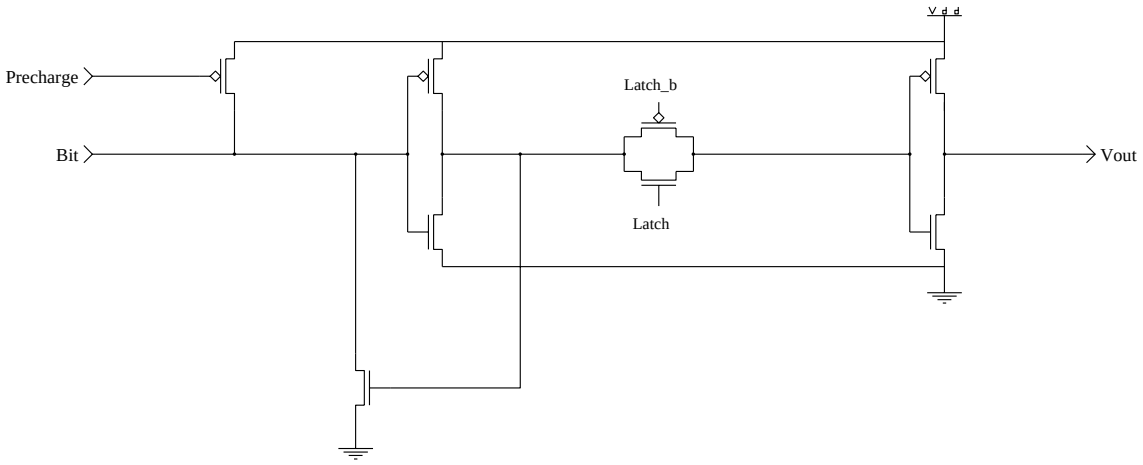


Figure 5.3: Sense Amplifier Schematic

performance of subthreshold MOS circuits vary as a function of temperature. However, the pixel block circuit can be temperature compensated using current biasing techniques similar to techniques used with bipolar transistors [70]. The following analysis is based on a fixed temperature.

5.3.1 One Bit A/D Converter

The one bit A/D converter was designed to minimize power while still operating at the desired sampling frequency of 1kHz. In order to determine the switching frequency of this circuit we will analyze the simplified small signal model of the cross coupled pair and the output amplifier shown in Figure 5.4. First, assume that the capacitance on the drain of **M8** is equal to the capacitance on drain **M7**. This assumption is false but it will simplify the analysis, and if we select the drain capacitance $c_1 = \max(c_{d8}, c_{d7})$ the analysis will produce an upper bound on the comparator speed. Next assume that the cross coupled comparator can be separated from the output amplifier by using the Miller capacitance approximation on **M11**. This will also produce an upper bound

on the comparators switching speed. Assume that the voltage difference between the gates of **M5** and **M6** is ϵ. Then we have

$$v_1(g_{o7} + g_{o5} + (c_{d7} + c_f)\frac{d}{dt}) - v_2(c_f\frac{d}{dt} - gm_7) = \epsilon\frac{gm_5}{2}, \tag{5.1}$$

and

$$v_2(g_{o8} + g_{o6} + (c_{d8} + c_f)\frac{d}{dt}) - v_1(c_f\frac{d}{dt} - gm_8) = -\epsilon\frac{gm_6}{2}. \tag{5.2}$$

Assume that $g_{o7} + g_{o5} = g_{o8} + g_{o6} = g_o$, $c_{d7} = c_{d8} = c_1$, $gm_7 = gm_8 = gm_n$, and $gm_5 = gm_6 = gm_p$. Note that c_1 should be set equal to all of the capacitance seen by the drain of **M8** including the Miller capacitance of **M11**. If we define $\delta v = v_2 - v_1$ then

$$\delta v(g_o - gm_n + (c_1 + 2c_f)\frac{d}{dt}) = -gm_p\epsilon \tag{5.3}$$

and

$$\delta v = \frac{gm_p\epsilon}{gm_n - g_o}e^{\frac{gm_n - g_o}{c_1 + 2c_f}t} \tag{5.4}$$

Now assume that δv must change by at least 400mV to switch the output. This implies that the output amplifier must have a gain of at least 25. We can determine the switching time of the latch, t_{latch}, from

$$t_{latch} = -\frac{c_1 + 2c_f}{gm_n - g_o}\ln(\frac{0.4v}{\frac{gm_p\epsilon}{gm_n - g_o}}). \tag{5.5}$$

Note that as ϵ tends to zero the bound on t_{latch} tends to infinity. This is the typical metastability problem with latches [30]. If we assume that the latch switches before the output amplifier switches then the speed of the output amplifier is slew rate limited. Therefore,

$$t_{amp} = \frac{c_{d11}V_{dd}}{I_{bias}} \tag{5.6}$$

and the total switching time for the one bit A/D is

$$t_{total} = t_{amp} + t_{latch}. \tag{5.7}$$

We determine $I_{bias} = I_{ds4} = I_{ds10}$ by selecting a minimum ϵ and then reducing I_{bias} until the switching time is less than approximately half of the sample period. For a switching speed greater than 0.1ms, with $\epsilon = 0.01v$, we selected a bias current of 5nA. The switching speed was verified using Hspice. If **M4** and **M10** are matched, then the maximum power dissipation, P_{AD}, of the comparator is

$$P_{AD} = 2I_{ds4}V_{dd}. \tag{5.8}$$

This corresponds to 50nW with a bias current $I_{ds4} = 5$nA and power supply voltage $V_{dd} = 5$V.

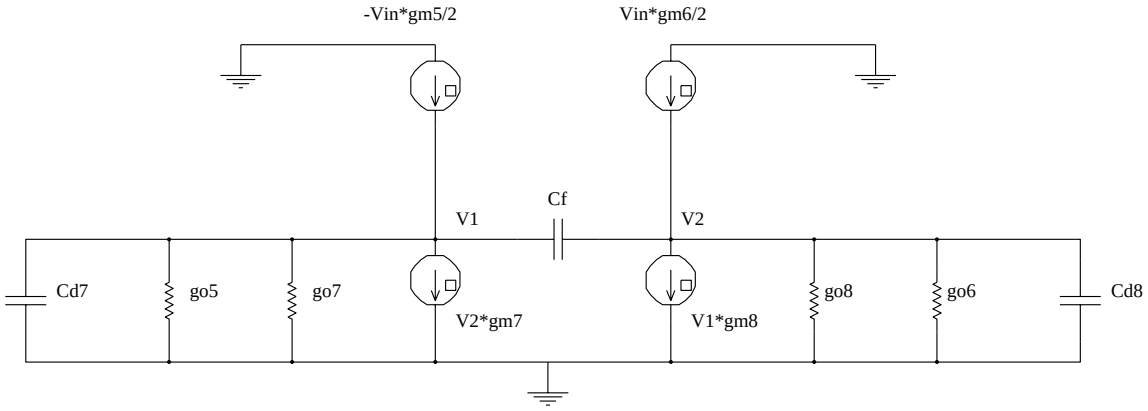


Figure 5.4: Small Signal Model of One Bit Latched A/D Converter

5.3.2 One Bit D/A Converter

Since fixed pattern noise, defined as the average pixel to pixel variation when the image sensor is uniformly illuminated, limits the performance of area image sensors we analyze the pixel to pixel feedback current variation caused by the one bit D/A

converter. The drain current I_{ds} through a subthreshold MOS transistor is

$$I_{ds} = \frac{W}{L} I_0 e^{\frac{(V_{sub}-V_s)(1-\kappa)}{V_t}} e^{\frac{\kappa V_{gs}}{V_t}} (1 - e^{-\frac{V_{ds}}{V_t}} + \frac{V_{ds}}{V_0}\frac{L_0}{L}). \tag{5.9}$$

This is derived in [45]. If we assume that channel length modulation can be neglected, $V_{sub} - V_s = 0$, and $V_{ds} \geq 4V_t$ so that $e^{-\frac{V_{ds}}{V_t}} \approx 0$ then the above equation simplifies to

$$I_{ds} = \frac{W}{L} I_0 e^{\frac{\kappa V_{gs}}{V_t}}. \tag{5.10}$$

I_0 is an exponential function of threshold voltage. Therefore, we can alter the above equation to

$$I_{ds} = \frac{W}{L} \hat{I}_0 e^{\frac{-\kappa V_{threshold}}{V_t}} e^{\frac{\kappa V_{gs}}{V_t}}. \tag{5.11}$$

This assumes that κ is not a function of threshold voltage. This assumption is true only as a first order approximation (see [45]). When the input to the one bit D/A is high the output current is

$$I_{outhigh} = I_{ds14} + I_{ds20}, \tag{5.12}$$

and when the input is low the output current is

$$I_{outlow} = I_{ds20} - e^{\frac{\kappa(V_{threshold19}-V_{threshold18})}{V_t}} I_{ds14}. \tag{5.13}$$

This assumes that threshold voltage (not device size) mismatch causes most of the error in the current mirror, **M18** and **M19** . The one bit first order sigma delta modulators gain, Δ, is determined by the difference between $I_{outhigh}$ and I_{outlow}

$$\Delta = I_{outhigh} - I_{outlow} = I_{ds14}(1 + e^{\frac{\kappa(V_{threshold19}-V_{threshold18})}{V_t}}). \tag{5.14}$$

Note that in Chapter 2 the difference between the two quantizer outputs was $2b$. Therefore, $\Delta = 2b$. If we are given the distribution of $V_{threshold}$, $f_v(v)$, and we assume that all threshold voltages are independent we can calculate the distribution of I_{ds},

$f_I(i)$, and the distribution of Δ, $f_\Delta(\delta)$. The distribution of I_{ds} given $f_v(v)$ is

$$f_I(i) = \frac{f_v(\ln(\frac{C}{i})}{|i|}, \tag{5.15}$$

where

$$C = \frac{W}{L}\hat{I}_0 e^{\frac{\kappa V_{gs}}{V_t}}. \tag{5.16}$$

Let

$$X = V_{threshold19} - V_{threshold18}, \tag{5.17}$$

$$Y = e^X, \tag{5.18}$$

and

$$Z = I_{ds14}Y. \tag{5.19}$$

Then

$$f_x(x) = \int_{-\infty}^{\infty} f_v(v+x)f_v(v)dv, \tag{5.20}$$

$$f_y(y) = \frac{f_x(\ln(y))}{y}, \tag{5.21}$$

and

$$f_z(z) = \int_{-\infty}^{\infty} f_I(\frac{z}{y})f_y(y)dy. \tag{5.22}$$

Therefore

$$f_\Delta(\delta) = f_I(\delta) * f_z(\delta). \tag{5.23}$$

Using the following assumed values,

- the absolute value of $V_{threshold}$ is uniformly distributed between 0.795V and 0.805V for both n and p devices,
- $I_{ds14} = I_{ds20} = 5$nA,
- $\kappa = 0.7$ for n and p devices,
- and $V_t = 0.026$V,

we have computed the histogram of Δ shown in Figure 5.5. This histogram was generated using a Monte Carlo simulation with 4096 data points. The standard deviation of Δ is 0.97nA. This shows that the one bit D/A converter causes an average pixel to pixel error of 9.7%. Therefore the fixed pattern noise caused by the D/A converter is 9.7%.

Noise generated by the one bit D/A converter limits the maximum achievable SNR of the sigma delta modulator, see Chapter 2. In order to analyze this limitation we will determine the one bit D/A converters noise by referring it to the output of the integrating capacitor **C1**. Using the simplified small signal circuits in Figures 5.6 and 5.7 we can determine the output referred noise power. If the input is low and assuming $g_o << gm$, $C_{dn} << C_1$, and $C_{dg18} << C_1$ then the noise transfer function for I_{n14} and I_{n19} is

$$DA_{n1}(s) = -\frac{gm_{18} - c_{dg18}s}{(g_{o18} + g_{o20} + C_1 s)(gm_{19} + (c_{d19} + c_{dg18})s) + (gm_{18} - c_{dg18}s)c_{dg18}s}, \tag{5.24}$$

and the noise transfer function for I_{n19} and I_{n20} is

$$DA_{n2}(s) = \frac{1}{g_{o18} + g_{o20} + C_1 s}. \tag{5.25}$$

Using superposition and assuming that the noise sources are independent and wide sense stationary, the output referred noise spectrum, when the input is low, is

$$N^2_{outputL}(f) = (I^2_{n14}(f) + I^2_{n19}(f))|DA_{n1}(j2\pi f)|^2 + (I^2_{n18}(f) + I^2_{n20}(f))|DA_{n2}(j2\pi f)|^2. \tag{5.26}$$

If the input is high, and using the previous assumptions, the noise transfer function for I_{n14} is

$$DA_{n3} = \frac{gm_{14}}{(g_{o17} + g_{o20} + C_1 s)(gm_{17} + c_{d14}s) - g_{o17}gm_{14}}, \tag{5.27}$$

the noise transfer function for I_{n17} is

$$DA_{n4} = \frac{-(gm_{17} + C_1 s)}{(g_{o17} + g_{o20} + C_1 s)(gm_{17} + c_{d14}s) - g_{o17}gm_{17}}, \tag{5.28}$$

and the noise transfer function for I_{n20} is

$$DA_{n5} = \frac{gm_{17} + c_{d14}s}{(g_{o17} + g_{o20} + C_1 s)(gm_{17} + c_{d14}s) - g_{o17}} \tag{5.29}$$

When the input is high the output referred noise spectrum is

$$N^2_{outputH}(f) = I^2_{n14}(f)|DA_{n3}(j2\pi f)|^2 + I^2_{n17}(f)|DA_{n4}(j2\pi f)|^2 + I^2_{n20}(f)|DA_{n5}(j2\pi f)|^2. \tag{5.30}$$

If we assume a duty cycle of 10% for **PHI1** and a duty cycle of 50% for the input then the total output referred noise is

$$N^2_{totalDA} = \frac{1}{20}\int_{-\infty}^{\infty}(N^2_{outputL}(f) + N^2_{outputH}(f))df \tag{5.31}$$

Using the following circuit values,

- **C1** = 650fF,
- I_{ds14} = 5nA,
- I_{ds20} = 5nA,
- temperature = 23 degree centigrade,
- early voltage = 25V for n and p devices,

and using the noise models in [17, 60] $N^2_{totalDA}$ is approximately 4.6×10^{-7}W. This is equivalent to 0.7mV of noise on **C1**. Therefore, the maximum SNR, in dB, is

$$SNR_{max} = 56.8 + 3\log_2(L). \tag{5.32}$$

Where L is the oversampling ratio of the sigma delta modulator. Assuming **M14** and **M20** are matched and the duty cycle of **PHI1** is 10% then the power dissipation, P_{DA}, of the one bit D/A is

$$P_{DA} = 0.2 I_{ds14} V_{dd} \tag{5.33}$$

This corresponds to 20nW at a bias current $I_{ds14} = 5$nA and a power supply voltage $V_{dd} = 3$V.

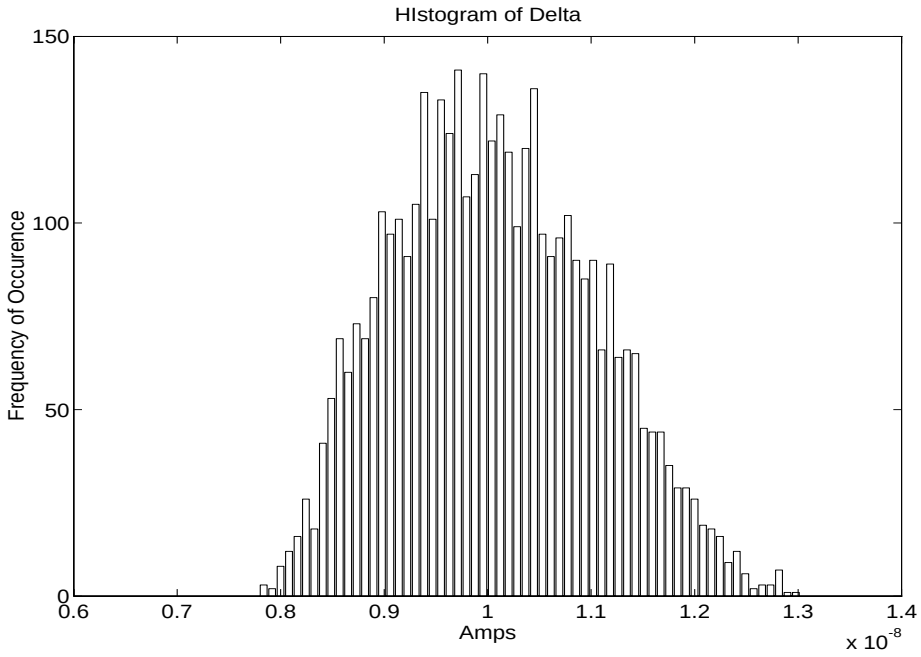


Figure 5.5: Histogram of Delta

5.4 Testing and Experimental Results

A 64×64 pixel block area image sensor was designed and fabricated to test the pixel block. The test chip was fabricated in a 1.2μm CMOS process with one layer of polysilicon and two layers of metal. Each pixel block contains 22 transistors and occupies 60μm×60μm. The photodiode in the pixel block occupies 105μm^2, this corresponds to a fill factor of 3%. The chip uses a 5V power supply, V_{dd}. The second layer of metal was used as a photo shield in order to reduce the chance of photo induced latch up in the pixel block circuits. Table 5.1 shows all of the characteristics of this area image sensor.

This sensor was tested using two different test beds. The first was for capturing still images, and the second for measuring SNR and dynamic range. A block diagram

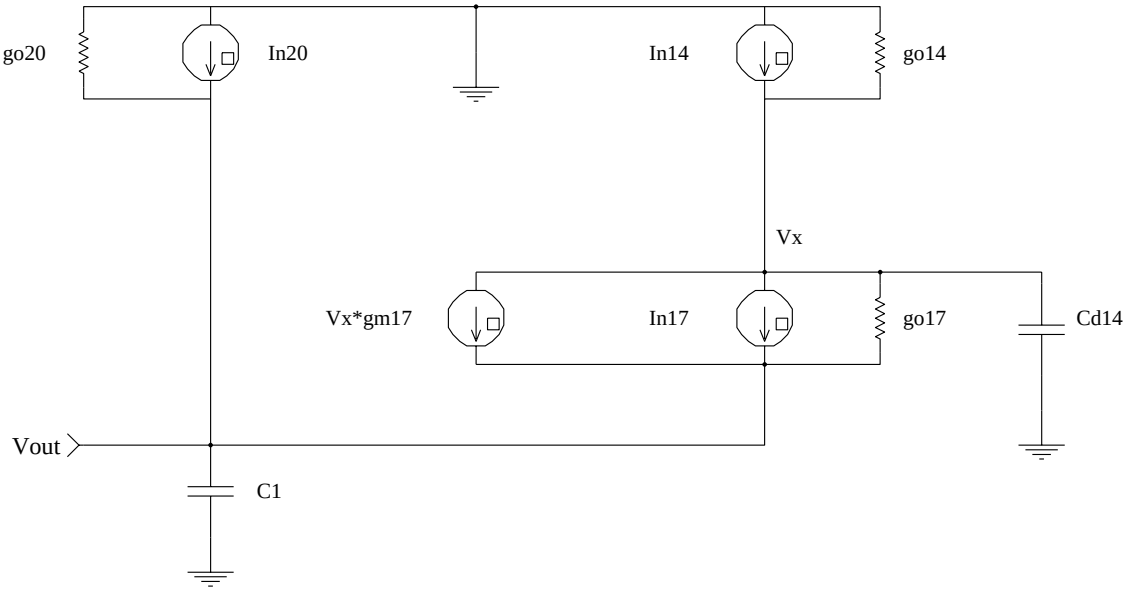


Figure 5.6: Small Signal Model of One Bit D/A Converter 1

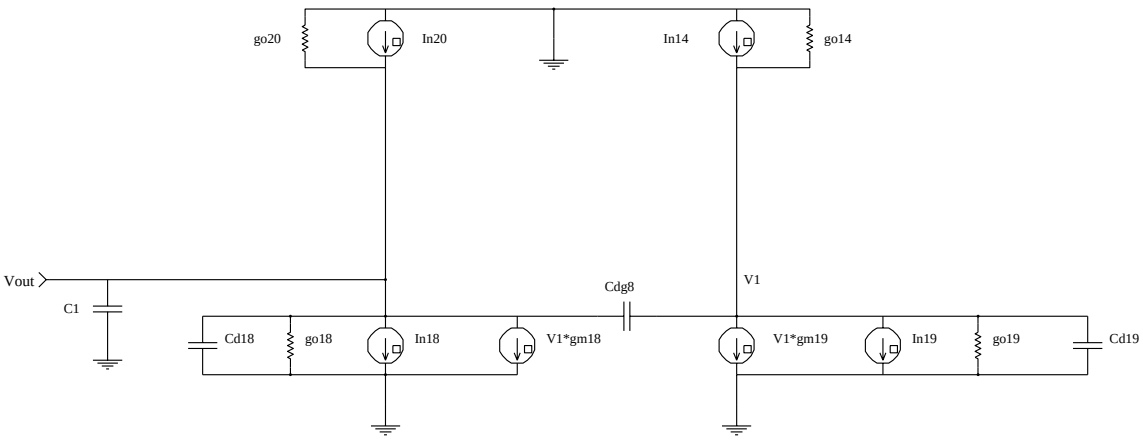


Figure 5.7: Small Signal Model of One Bit D/A Converter 2

Main Characteristics of 64x64 Area Image Sensor	
Technology	$1.2\mu m$, 2-layer metal, 1-layer poly, nwell CMOS
Die Area	$6500\mu m \times 5000\mu m$
Pixel Area	$60\mu m \times 60\mu m$
Number of transistors per pixel	22
Phototransistor Area	$105\mu m^2$
Fill Factor	3%
Package	84pin PGA
Supply Voltage	5v
SNR	61dB
Dynamic Range	93dB
Fixed Pattern Noise	$\approx$ 10% RMS
Power Dissipation per pixel	$<$ 60nW

Table 5.1: Area Image Sensor Characteristics - measured at 23 degrees centigrade

of the first test bed is shown in Figure 5.8 [38]. The ASIC is an ACTEL 1010 FPGA. It was programmed to produce clocking signals for the image sensor and store the image data in SRAMs located on the same pc-board. Image data was read from the SRAMs into the PC using the ISA bus interface. The image data was then decimated and displayed on the PC monitor. Decimation was performed in software using a triangular FIR filter, i.e. a sinc squared filter. The test chip was clocked at 1kHz and 32 bit planes were used to generate each image. Figure 5.9 shows a scan from a 35mm print (left), and the image obtained by the sensor (right) when contact exposed to the 35mm negative. Contact exposure is the process of placing a negative on the chip surface and shining light through it. This method was used because we lacked the necessary optical equipment used in most image sensor applications, i.e. a mechanically stable lens and a dark room. The fixed pattern noise was estimated to be $\approx$ 10% by averaging 36 adjacent pixels values from the image in Figure 5.9. These 36 pixels were taken from a region of the image where the illumination was expected to be uniform. The fixed pattern noise was caused by both beta variation in the phototransistors, see Chapter 3 and variations in the one bit D/A converter.

The second test bed is shown in Figure 5.10. Using a light emitting diode and an op-amp we generated a sinusoidal light source and shined it onto the sensor chip. Figure 5.11 shows the output from a single pixel. Approximately 65000 samples were

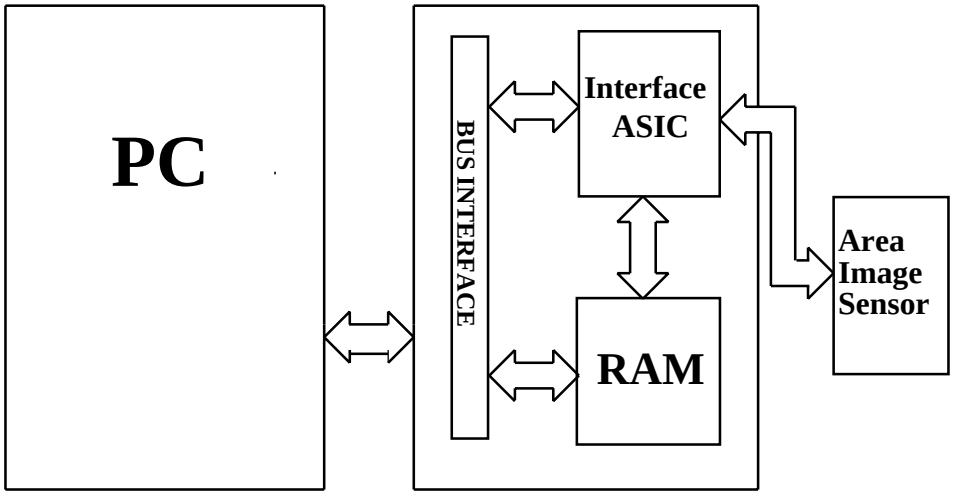


Figure 5.8: Test Setup For Imaging

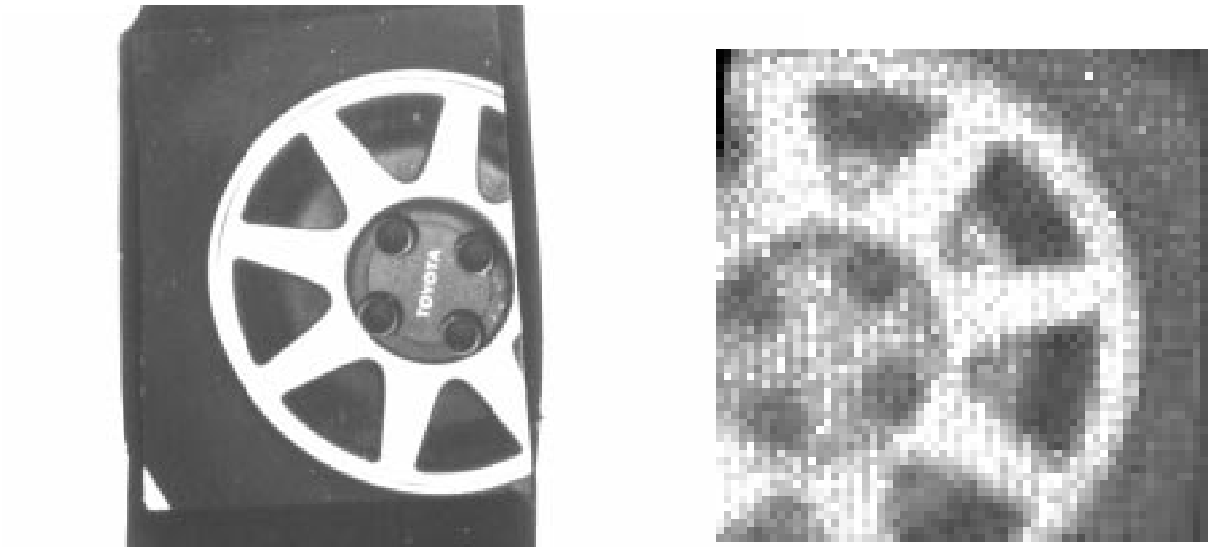


Figure 5.9: 300dpi scan of print from negative (left). 64×64 image produced by sensor using 35mm negative contact exposure (right).

measured from eight pixels. Each pixel block was again clocked at 1kHz.

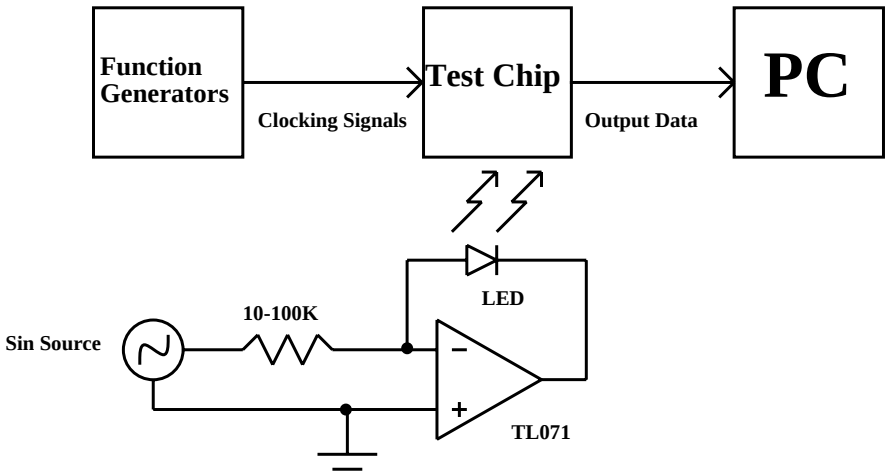


Figure 5.10: Test Setup For SNR and Dynamic Range Measurement

A maximum SNR, defined as the signal to quantization noise ratio, of 61dB was measured with a shutter duty cycle of 100%. This was determined using the power spectral density plots shown in Figures 5.12 and 5.13. These two graphs show the power spectral density of one pixel block with a 0.1Hz sinewave input. The power spectral density was calculated using the Matlab function "spectrum" with a window size of 8192 and 8 overlapping blocks. The decimated output from the sensor is shown in Figure 5.14. A 8192 tap low pass FIR filter with a bandwidth of approximately 0.2Hz was used for decimation. SNR degradation due to charge injection of the digital circuitry in close proximity to the analog sensors is negligible since the frequency of operation is 1kHz. The dynamic range of the sensor was determined to be at least 93dB. This was calculated by multiplying the inverse of the minimum shutter duty cycle by the measured SNR. The minimum shutter duty cycle was measured to be approximately $\frac{1}{1000}$ and the SNR at that shutter setting is 33dB. If the shutter duty cycle was reduced below this value the data was distorted. Each pixel block consumes

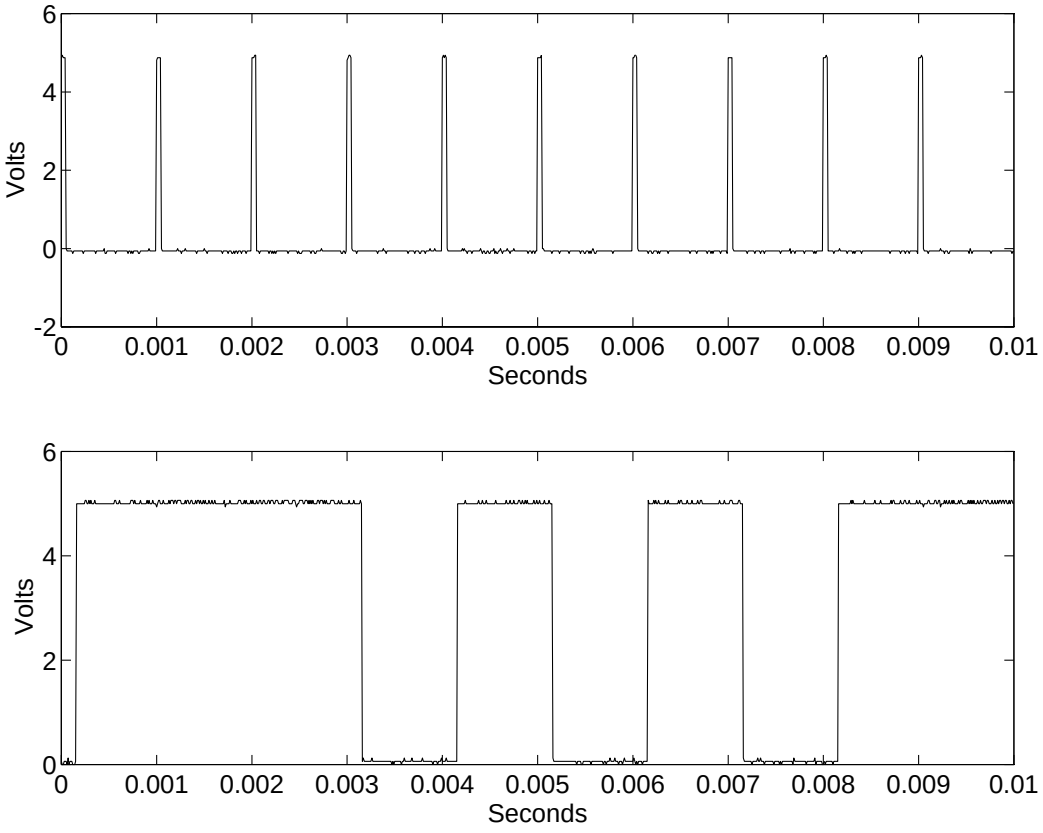


Figure 5.11: Single pixel Sigma-Delta modulator output from HP54601A. The top graph is **PHI2** versus time and the bottom graph is the pixel output waveform versus time.

less than 60nW. This was determined by measuring the total power dissipation of the chip, subtracting the calculated power dissipation of the pads, and finally dividing the remaining power by the number of pixels in the array, 4096.

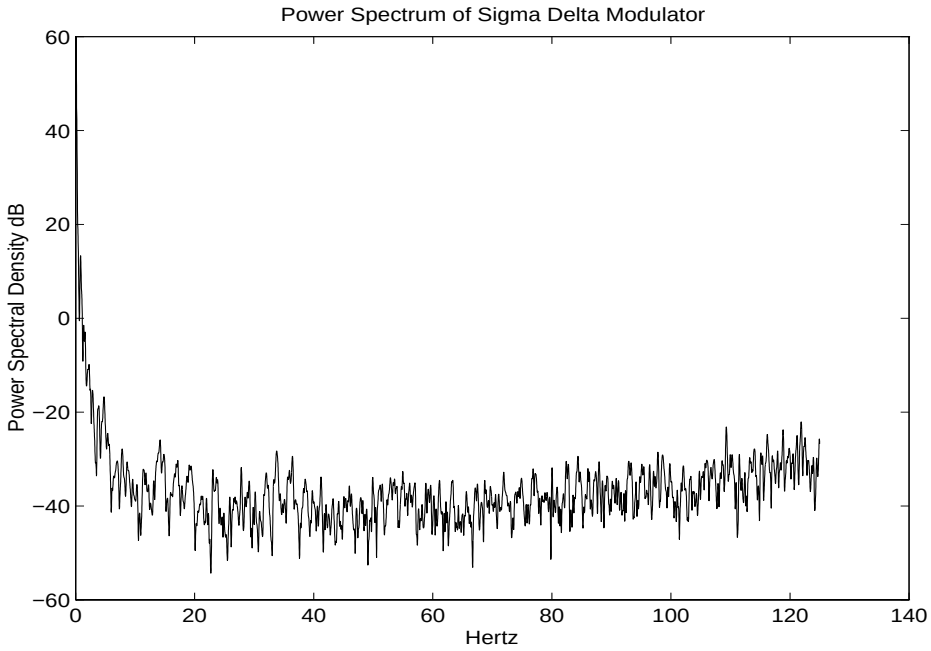


Figure 5.12: Power Spectral Density of Pixel Block Sigma Delta Modulator with 0.1Hz Sinewave Input - Shutter Duty Cycle = 100%

5.5 Summary and Conclusions

A circuit level description of our original pixel block has been presented. This circuit was tested using a 64×64 pixel block area image sensor. Results show that this circuit dissipates very little power and achieves high SNR and dynamic range. The pixel block circuit also has several limitations including, large size, low fill factor, and large fixed pattern noise. Table 5.2 highlights some of the differences between this sensor and a comparable CCD sensor [51]. Note that pixel size, fill factor and fixed pattern noise are areas that must be improved in future designs for this technology to become competitive with CCDs.

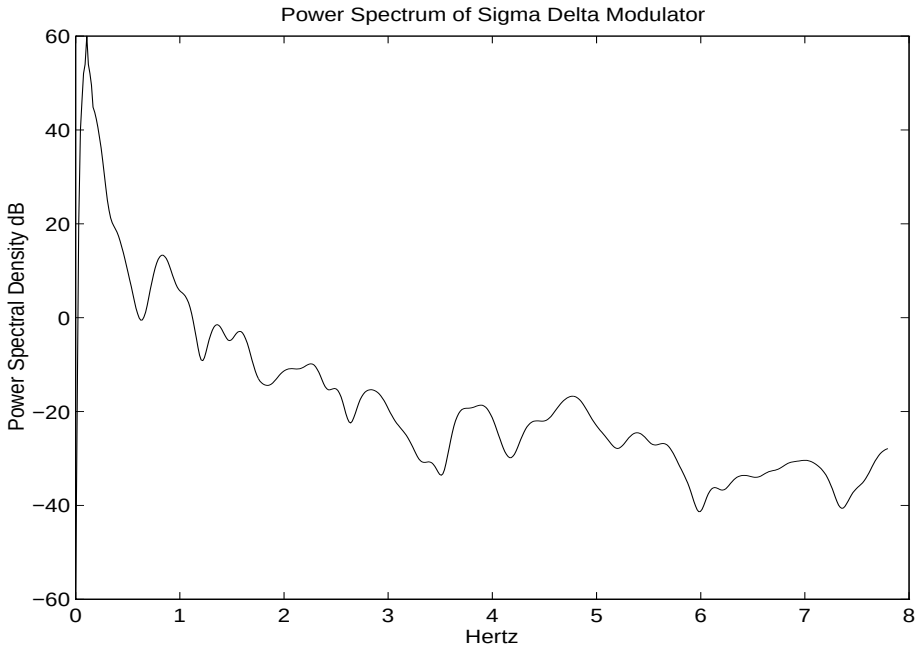


Figure 5.13: Power Spectral Density Pixel Block Sigma Delta Modulator with 0.1Hz Sinewave Input - Shutter Duty Cycle = 100%

Parameter	CCD	Original Pixel Block
Technology	1.2 μm	1.2 μm
Transistors/pixel	1	22
Pixel Area	$11\mu m \times 8\mu m$	$60\mu m \times 60\mu m$
Fill Factor	30%	3%
SNR	72dB	61dB
Dynamic range	72dB	92dB
Power Dissipation/pixel	20μW	$<$ 60nW
Fixed Pattern Noise	$< 1\%$	$\approx 10\%$

Table 5.2:

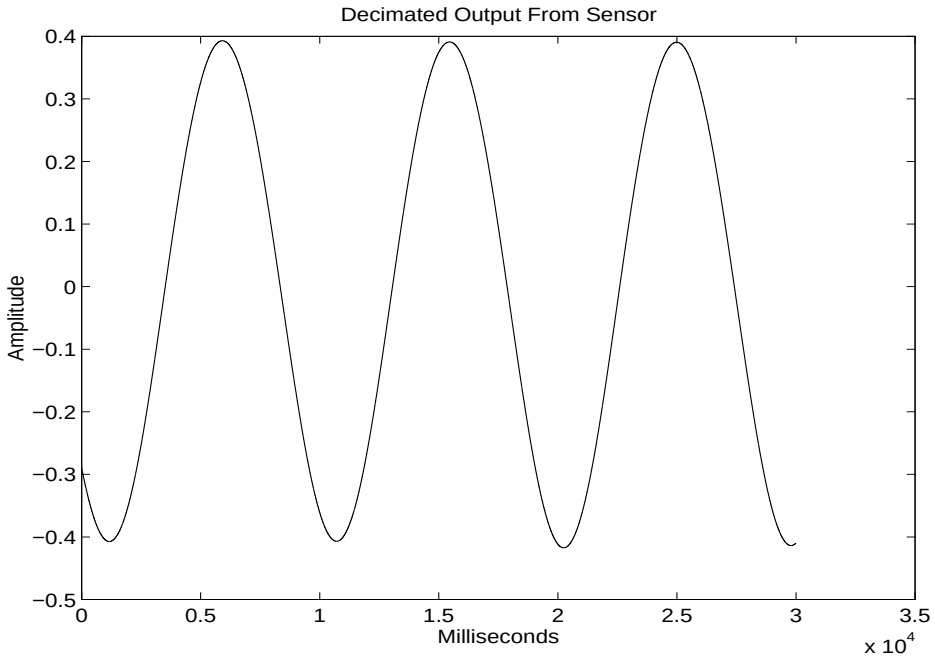


Figure 5.14: Decimated Output from a Pixel Block Using a 8192 Tap Low Pass FIR Filter

This design was intended to show the feasibility of pixel-level A/D conversion in a CMOS area image sensor, but it was not properly designed for complete testability. We needed to test both the sensor performance and the sigma delta modulator performance. The separation of these two measurements proved to be beyond the capabilities of the available instruments. For example, to test the sigma delta modulator independently from the sensor we need to generate precision time varying currents between 10pA and 10nA. Additionally, currents on the order of 1pA must be measured to test the phototransistors dark current and other low light characteristics. Leakage currents in the one bit D/A converter are larger than the phototransistor leakage currents we needed to measure. This limited our testing to the measurement of the sigma delta modulator performance. We were not able to directly measure the characteristics of the phototransistors in the image sensor.

Chapter 6

Improved Pixel Block Circuit Design

6.1 Introduction

In the previous chapter we discussed the first pixel block circuit and a 64×64 CMOS area image sensor chip. The test chip identified several pixel block circuit limitations including, large pixel size, low fill factor, and high fixed pattern noise. In this chapter we describe an improved pixel block circuit. This circuit was designed to reduce pixel size, increase fill factor, reduce fixed pattern noise, and retain all of the benefits of the original pixel block circuit, i.e. wide dynamic range and low power dissipation. The first section describes the operation of the pixel block circuit. The second section analyzes its performance and discusses design trade-offs. Differences between the improved pixel block and the original pixel block are highlighted. The following section presents results from a 4×4 pixel block image sensor chip. Finally we discuss the circuit's advantages over the original pixel block circuit and its limitations by comparing it with a comparable CCD. Future directions for this work are also presented.

6.2 Circuit Description

The improved pixel block circuit, with transistor sizes, is shown in Figure 6.1. Notice that we have replaced the phototransistor in the previous pixel block with a photodiode in order to reduce fixed pattern noise. The photodiode was constructed by diffusing a p+ region into an nwell. The nwell voltage is fixed by V_{dd} and the p+ diffusion is connected to the shutter circuitry **M1** and **M2**. **D1** operates in charge skimming mode, i.e. the photodiode has a constant voltage bias ($V_{dd} - V_{bias2}$). The shutter is controlled by **SHUTTER** and **SHUTTERB**. When **SHUTTER** is low and **SHUTTERB** is high current from **D1** flows through **M1** into **Vbias2**. This stops the input photocurrent from flowing into the sigma delta modulator. When **SHUTTER** is high and **SHUTTERB** is low photocurrent from **D1** flows through **M2** and into the sigma delta modulator. Photocurrent from **D1** is integrated on **C1** using an active integrator. The active integrator is formed using a transconductance amplifier with a feedback capacitor, **C1**. **C1** is connected between the negative input of transconductance amplifier and its output. Photocurrent from **D1** causes the output voltage of the integrator to decrease.

The output voltage of the active integrator is quantized using a latched comparator. This comparator operates exactly like the comparator used in the original pixel block circuit. The quantized output from the one bit A/D is fed back to the gate of **M16**. The stack of transistors **M16**, **M17**, and **M18** form a switched capacitor circuit that acts as a one bit D/A converter. When **PHI2** is high **C2** is shorted to ground. If the gate of **M16** is high and **PHI1** is high **C2** is shorted to the input of the integrator. This removes charge from **C1** and increases the output voltage of the integrator. The amount of charge, q, removed from **C1** is

$$q = V_{bias2} C_2, \tag{6.1}$$

and this increases the integrators output voltage, ΔV_{out}, by

$$\Delta V_{out} = \frac{C_2}{C_1} V_{bias2}. \tag{6.2}$$

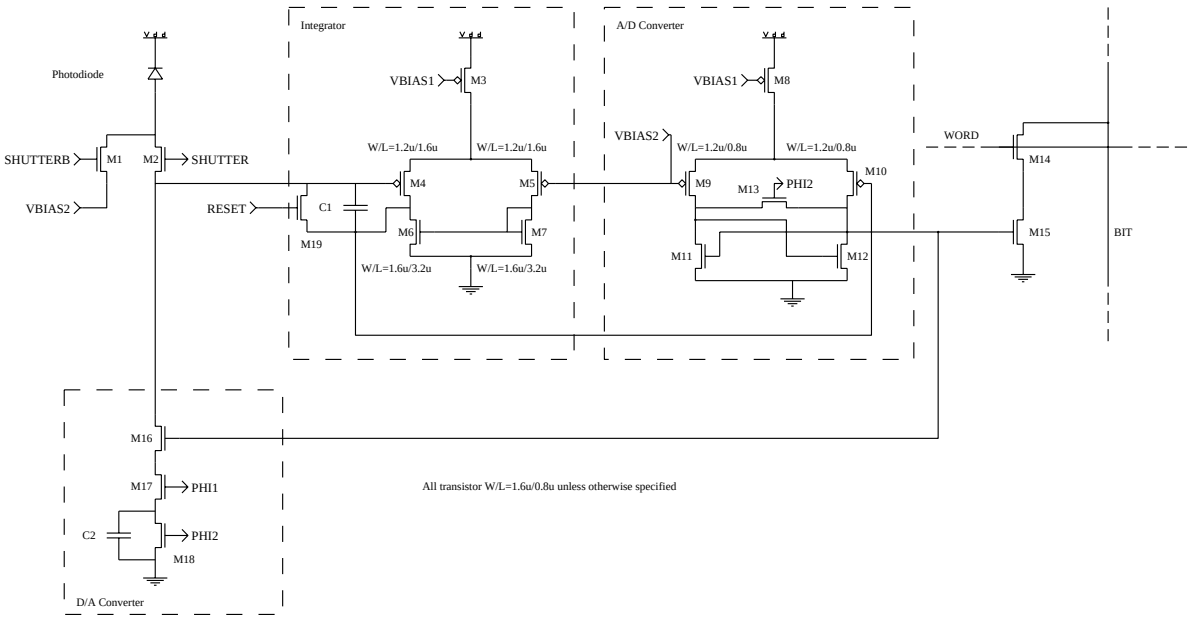


Figure 6.1: Improved Pixel Block Schematic

If the gate of **M16** is low charge is not transferred from **C1** to **C2**. **C2** was designed for a maximum photocurrent of 20pA at a sampling rate of 1kHz. This implies that **C2** must be greater than or equal to 13.3fF given $V_{bias2} = 1.5$V. We designed **C2** to be 20fF. **C1** was design to be 100fF so that $\Delta V_{out} = 300$mV. The control and output circuitry are exactly the same as the original pixel block circuit design.

6.3 Circuit Analysis

Again, all of the MOS circuits used in the pixel block are operated below threshold in order to reduce power dissipation, and increase voltage gain [70]. As discussed in the previous chapter, the characteristics and performance of subthreshold MOS circuits vary as a function of temperature. We will analyze the pixel block circuit at room temperature. We will also omit temperature variation analysis because the pixel block circuit can be temperature compensated using current biasing techniques similar to techniques use by bipolar transistors [70].

6.3.1 Active Integrator

A sigma delta modulator's SNR is limited by the performance of its integrator (see Chapter 2). We will analyze effects such as, finite DC gain, slew rate limiting, and device noise, in order to determine how they limit the SNR of our pixel block circuit. We will begin by analyzing the finite DC gain, gain bandwidth product, and noise of the transconductance amplifier using the small signal model shown in Figure 6.2. The amplifier is biased below threshold in order to conserve power and maximize DC gain [70]. The DC open loop gain, A_{DC} is

$$A_{DC} = \frac{gm_4}{g_{o4} + g_{o6}}. \tag{6.3}$$

The output impedance, R_{output}, is

$$R_{output} = \frac{1}{g_{o4} + g_{o6}}, \tag{6.4}$$

and the dominant output pole, BW, is

$$BW = \frac{1}{2\pi C_L R_{output}}. \tag{6.5}$$

Using the following circuit parameters,

- **C1** = 100fF,
- $I_{ds3} = 5$nA,
- $\kappa = 0.7$ for both n and p devices,
- $V_t = 0.026$V,
- early voltage = 25V for both n and p devices

the finite DC gain of the amplifier is 350, and the output impedance is 2.5G ohms. Using the analysis in Chapter 2, the DC gain of the amplifier will reduce the total SNR of the sigma delta modulator by less than 1dB if the oversampling ratio is below

350. The gain bandwidth product of the amplifier is 110kHz. Once again using the small signal model in Figure 6.2 the input referred noise of the transconductance amplifier is approximately

$$N^2_{input}(f) = (I^2_{n4}(f) + I^2_{n6}(f))gm_4^{-2}. \tag{6.6}$$

This assumes that the noise sources are independent and wide sense stationary.

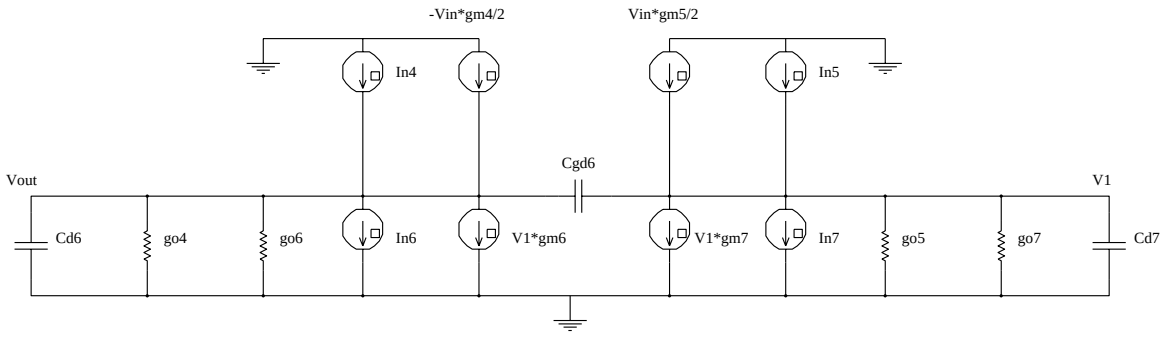


Figure 6.2: Small Signal Model of Tranconductance Amplifier

Large signal performance of the transconductance amplifier can also limit the SNR of the sigma delta modulator. Specifically, we will analyze output voltage clipping and slew rate limiting of the amplifier. The total linear output range of the amplifier is set by the device saturation of **M4** and **M6**. Under normal operation the gates of **M4** and **M5** should be at the approximately the same voltage. Therefore, the total linear output range of the amplifier is approximately $4V_t$ to $V_{bias2} + |V_{threshold4}| - 4V_t$. If the output voltage of the amplifier leaves this linear range the gain of the amplifier will drop sharply, causing excess quantization noise to leak into the pass band. The maximum required output current for the transconductance amplifier is

approximately 20pA. This guarantees that the transconductance amplifier will not be slew rate limited at a bias current $I_{ds3} = 5$nA. A slew rate limited amplifier causes nonlinear distortion which limits the sigma delta modulator's performance [6]. The power dissipation of the integrator, P_{int}, is

$$P_{int} = I_{ds3}V_{dd}. \tag{6.7}$$

This corresponds to 15nW at a bias current of 5nA and a power supply voltage of $V_{dd} = 3$V.

6.3.2 One Bit A/D Converter

The one bit A/D converter's speed must be greater than the circuit's sampling rate, 1kHz. Since the comparator used in this design is almost identical to the original pixel block circuit, we can use the analysis of that circuit to guide this design. The only difference between the two comparators is that this design does not use an output gain stage. This affects output swing, speed, and power dissipation. The output swing will be reduced such that the minimum output is 0V and the maximum is $V_{bias2} + |V_{threshold4}|$. The comparators speed will decrease due to the increased capacitive loading on the drain of **M10**. Using HSPICE on the two comparators, with identical biases, the average propagation delay increased by 10%. Therefore, I_{ds8} was biased at 5nA. The maximum power dissipation, P_{AD}, will be reduced to

$$P_{AD} = I_{ds8}V_{dd}. \tag{6.8}$$

This corresponds to 15nW with a bias current I_{ds8} of 5nA and a power supply voltage $V_{dd} = 3$V.

6.3.3 One Bit D/A Converter

Fixed pattern noise limits the performance of area image sensors. The main source of fixed pattern noise in this circuit is pixel to pixel variation of the one bit D/A converter. The output magnitude of the one bit D/A converter is determined by the

size of **C2**. **C2**'s size is controlled by lithography, oxide growth and diffusion. If **C2** is sufficiently large fixed pattern noise can be reduced below 1%. This is confirmed by results from various test chips in our laboratory. Note that capacitors can be matched much closer than transistors operating below threshold [43, 70]. This implies that the improved pixel block circuit should have much lower fixed pattern noise than the original pixel block circuit.

Circuit noise limits the maximum achievable SNR of a sigma delta modulator (see Chapter 2). Circuit noise in the improved pixel block comes from two sources. The first is the transconductance amplifier and the second is the switched capacitor D/A circuitry. In order to maintain consistency with the noise analysis of the original pixel block circuit we will refer all of the system noise to the input of the one bit A/D. Using the two simplified small signal models in Figures 6.3 and 6.4, and the noise analysis of the active integrator the total system noise can be determined. The two small signal circuits separate the noise into independent components generated during **PHI1** and **PHI2**. Noise can further be classified by whether it is sampled or continuous. It is assumed that all of the noise sources considered in this analysis are independent. During **PHI2** **C1** samples the wide band noise from **M18** and aliases all of the power into the passband [35]. This results in a noise power at the output of transconductance amplifier equal to

$$N^2_{samplePHI1} = \frac{mkTC_2}{C_1^2}, \tag{6.9}$$

where m is the duty cycle of **PHI1** and **PHI2**. While **PHI1** is low, noise from the transconductance amplifier also contributes to the noise power at the input of the A/D. If a single pole approximation is assumed for the transconductance amplifier, then the total noise power during this time is

$$N^2_{continuousPHI1} = \int_{-\infty}^{\infty} \frac{N^2_{input}(f)}{1 + (2\pi(1 + \frac{C_2}{C_1})\frac{R_{output}C_1}{A_{DC}}f)^2} df \tag{6.10}$$

During **PHI1** noise from the transconductance amplifier, **M16** and **M17** are sampled

by **C1** and aliased back into the pass band. The noise power sampled onto **C1** is

$$N^2_{samplePHI2} = \int_{-\infty}^{\infty} \frac{m(\frac{C_2}{C_1})^2(V^2_{n16}(f)+V^2_{n17}(f))}{1+(2\pi\frac{R_{output}C_1(C_1+C_2)}{A_{DC}}C_1 f)^2}df + \tag{6.11}$$

$$\int_{-\infty}^{\infty} \frac{m(\frac{C_2+C_1}{C_1})^2 N^2_{input}(f)}{1+(2\pi\frac{R_{output}C_1(C_1+C_2)}{A_{DC}C_1}f)^2}df. \tag{6.12}$$

$$\tag{6.13}$$

The total noise seen at the output of the transconductance amplifier is the sum of all the noise components.

$$N^2_{total} = N^2_{samplePHI1} + N^2_{continuousPHI1} + N^2_{samplePHI2}. \tag{6.14}$$

Using the following circuit parameters,

- **C1** = 100fF,
- **C2** = 20fF,
- I_{ds3} = 5nA,
- $\kappa = 0.7$ for both n and p devices,
- V_t = 0.026V,
- early voltage = 25V for both n and p devices,

and the noise models in [17, 60], N^2_{total} = 65nW. This represents a noise voltage of 255μV on **C1**. Note that the total noise has been reduced when compared with our original pixel circuit. This is a result of the feedback in the transconductance amplifier reducing the output referred MOS device noise, and the low ratio of **C2** to **C1** reducing the switched capacitor noise. Therefore, the maximum SNR, in dB, of the sigma delta modulator is limited to

$$SNR_{max} = 51.4 + 3\log_2(L), \tag{6.15}$$

where L is the oversampling ratio of the sigma delta modulator. The maximum power dissipation of the one bit D/A, P_{DA}, is approximately

$$P_{DA} = C_2 V_{bias2} f_{sample} \tag{6.16}$$

Where f_{sample} is the sampling frequency of the sigma delta modulator. This assumes that all of the parasitic capacitances in the one bit D/A converter are much smaller than **C2**. This corresponds to 30pW with **VBIAS2** = 1.5V, C_2 = 20fF, and f_{sample} = 1kHz.

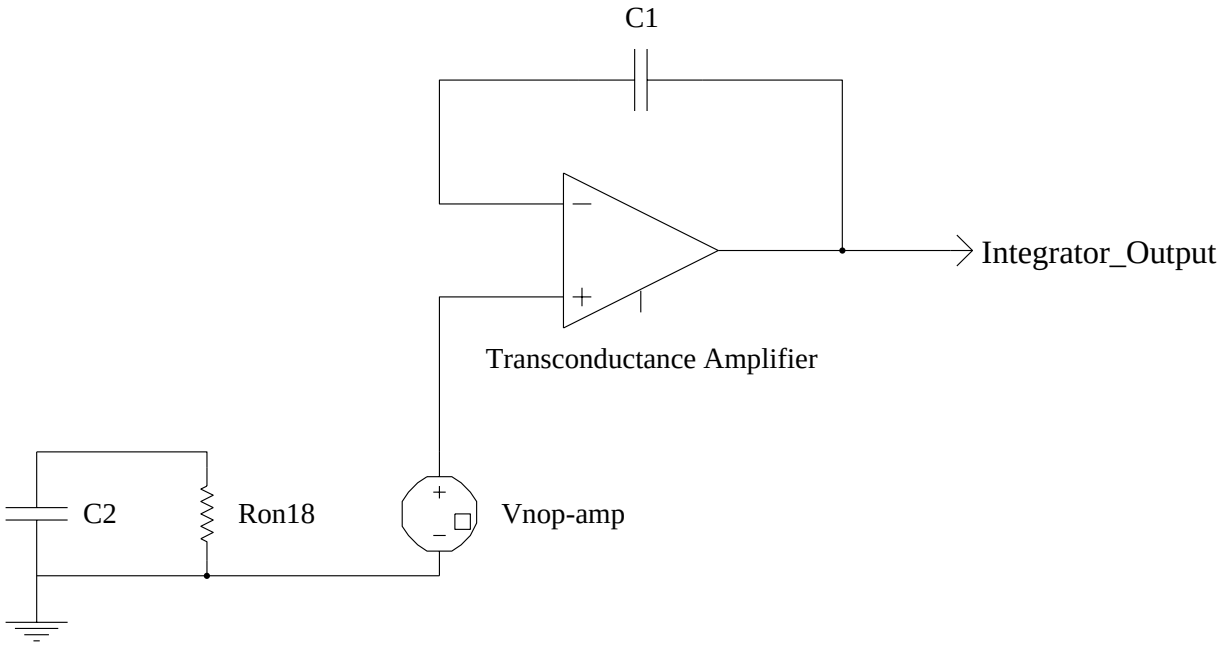


Figure 6.3: Small Signal Model of One Bit D/A Converter 1

6.4 Testing and Experimental Results

A 4×4 pixel block area image sensor was designed and fabricated to test the pixel block. The test chip was fabricated in a 0.8μm CMOS process with one layer of polysilicon and three layers of metal. Each pixel block contains 19 transistors and

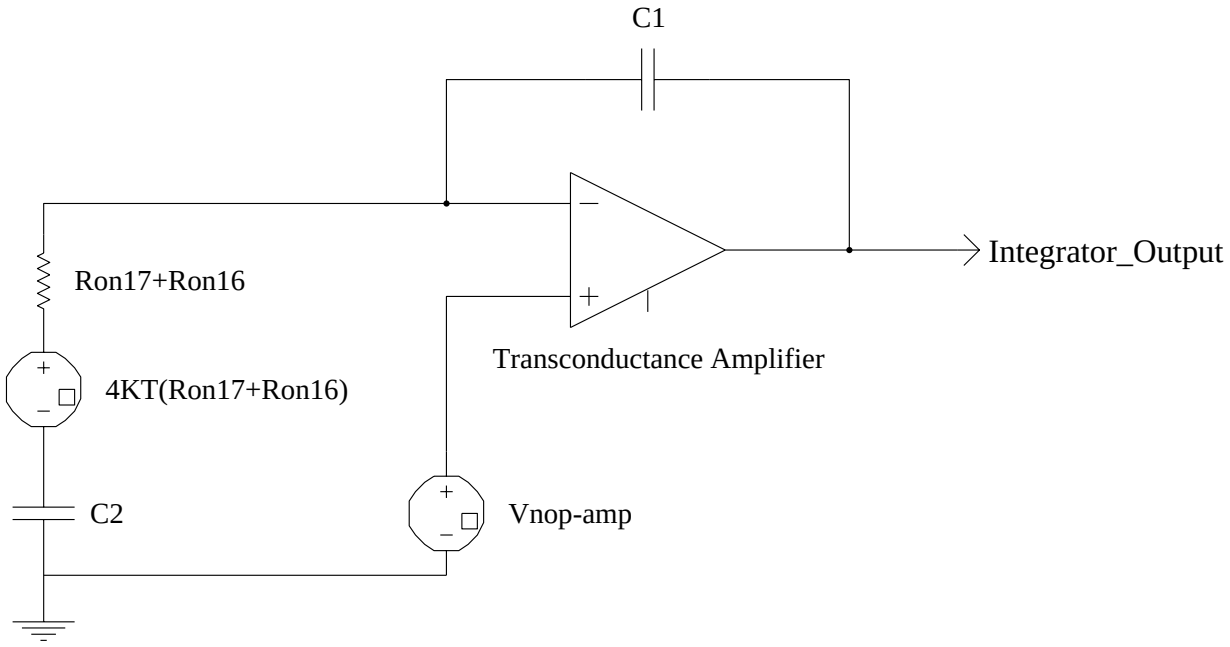


Figure 6.4: Small Signal Model of One Bit D/A Converter 2

occupies 30μm$\times 30\mu$m. The photodiode in the pixel block occupies 126μm^2, which corresponds to a fill factor of 14%. The chip uses a 3V power supply, V_{dd}. The third layer of metal was used as a photo shield in order to reduce the change of photo induced latch up in the pixel block circuits. Table 6.1 shows all of the characteristics of this area image sensor.

This sensor chip was tested using the test bed show in Figure 5.10. Using a light emitting diode and an op-amp we illuminated the sensor chip with both sinusoidal and DC light sources. Then approximately 65000 samples were measured from all 16 pixels. Each pixel block was clocked at 1kHz.

A maximum SNR of 52dB was measured with a shutter duty cycle of 100%. This was determined using the power spectral density plot shown in Figure 6.5. This graph shows the power spectral density of one pixel block with a 1.9Hz sin wave input. The power spectral density was calculated using the Matlab function "spectrum" with a window size of 8192 and 8 overlapping blocks. The decimated output from the sensor is shown in Figure 5.14. A 8192 tap low pass FIR filter with a bandwidth

Main Characteristics of 4×4 Area Image Sensor	
Technology	$0.8\mu m$, 3-layer metal, 1-layer poly, nwell CMOS
Die Area	$2100\mu m \times 2200\mu m$
Pixel Area	$30\mu m \times 30\mu m$
Number of transistors per pixel	19
Photodiode Area	$126\mu m^2$
Fill Factor	14%
Package	40pin DIP
Supply Voltage	3v
SNR	52dB
Dynamic Range	85dB
Fixed Pattern Noise	$\approx$ 1% RMS
Power Dissipation per pixel	$<$ 50nW

Table 6.1: Area Image Sensor Characteristics - measured at 21 degrees centigrade

of approximately 2Hz was used for decimation. Using the analysis in Chapter 2 and the previous section we believe that the SNR in this design was limited, when compared with the original pixel block design, by device noise, i.e. circuit noise from the transconductance amplifier and the switched capacitor one bit D/A converter. The dynamic range of sensor was determined to be at least 85dB. This was calculated by multiplying the inverse of the minimum shutter duty cycle by the measured SNR. The minimum shutter duty cycle was measured to be approximately $\frac{1}{100}$ and the SNR at that shutter setting is 45dB. If the shutter duty cycle was reduced below this value the data was distorted. A fixed pattern noise of approximately 1% was measured using a DC light source. The response of all 16 pixels were averaged to determined the fixed pattern noise. Each pixel block consumes less than 50nW. This was determined by measuring the total power dissipation of the chip, subtracting the calculated power dissipation of the pads, and finally dividing the remaining power by the number of pixels in the array, 16.

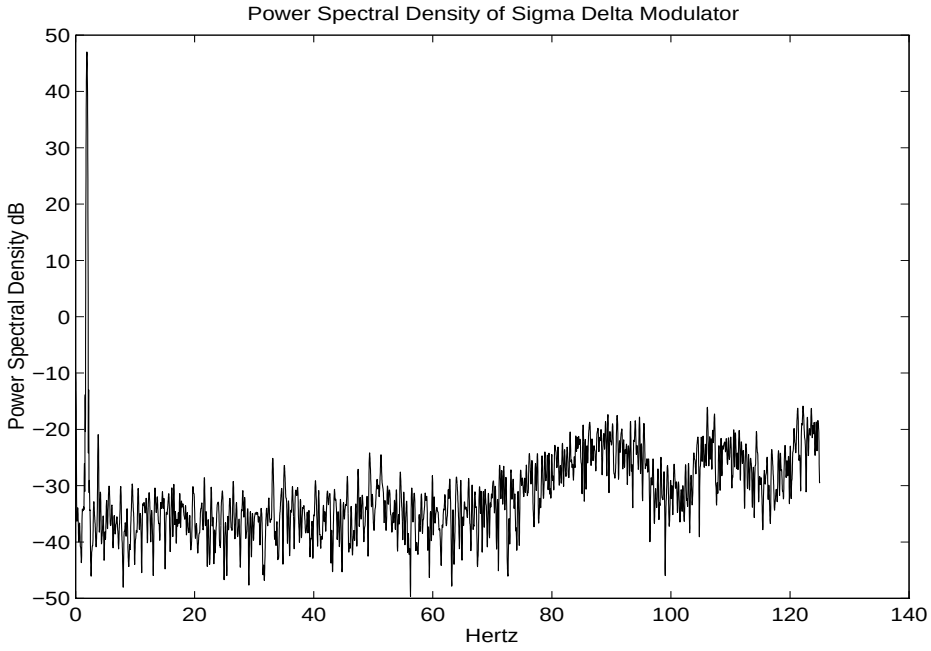


Figure 6.5: Power Spectral Density of Pixel Block Sigma Delta Modulator with 1.9Hz Sinewave Input - Shutter Duty Cycle = 100%

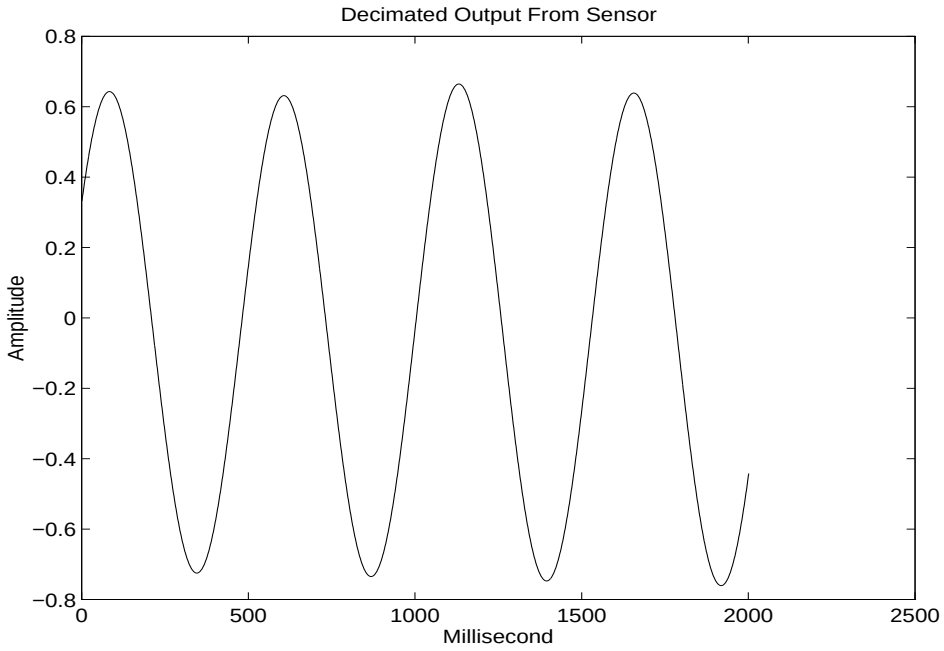


Figure 6.6: Decimated Output from a Pixel Block Using a 8192 Tap Low Pass FIR Filter

6.5 Summary and Future Work

A circuit level description of our improved pixel block circuit has been presented. This circuit was tested using a 4×4 pixel block area image sensor. Results show that this circuit dissipates very little power and achieves high dynamic range. The improved pixel block circuit still has a few limitations including, large size, and low fill factor. Table 6.2 highlights some of the differences between this sensor, our original pixel block circuit sensor, and a comparable CCD sensor [7]. Note that we have reduced the pixel size, increased the fill factor, reduced the fixed pattern noise, and reduced the number of transistors per pixel block when compared with the original pixel block circuit. When compared with a CCD pixel we still need to reduce the pixel size and increase the fill factor.

Parameter	CCD	O Pixel Block	I Pixel Block
Technology	0.8 μ m	1.2 μ m	0.8 μ m
Transistors/pixel	1	22	19
Pixel Area	$6.9\mu m \times 12.6\mu m$	$60\mu m \times 60\mu m$	$30\mu m \times 30\mu m$
Fill Factor	30%	3%	14%
SNR	72dB	61dB	52dB
Dynamic range	72dB	92dB	85dB
Power Dissipation/pixel	20μW	$<$ 60nW	$<$ 50nW
Fixed Pattern Noise	$<$ 1%	$\approx$ 10%	1%

Table 6.2: CCD to CMOS Pixel-Level A/D Sensor Comparison

We were not able to measure the phototransistor referred electron noise in the original pixel block circuit. Therefore, the improved pixel block circuit was designed to alleviate this problem. We were expecting the RMS photodiode referred electron noise to be 100fA, but it was less than 1fA. Parasitic p-n junctions in the one bit D/A converter generated an input current of 20fA. This made photodiode referred noise measurements impossible! In future pixel block designs all leakage currents must be keep below 0.1fA. This implies that all of the MOS devices inside the one bit D/A converter must have very low leakage currents. To achieve this goal the source drain junction sizes must be minimized and device thresholds must be high. This will allow accurate photodiode referred noise measurements to be performed.

Recently, several new pixel block designs have been investigated. Specifically, a pixel block circuit that multiplexes one A/D and one D/A between four adjacent photodiodes has been developed. A block diagram of this circuit is shown in Figure 6.7. This circuit has allowed us to reduce the pixel size to $11\mu m \times 11\mu m$ and increase the fill factor to 30% in a $0.5\mu m$ CMOS process with one layer of polysilicon and three layers of metal. There are only four transistors per pixel, but the clocking circuitry has become more complicated. This circuit is designed to scale with a standard CMOS process, and in a $0.25\mu m$ CMOS process the multiplexed pixel block will occupy less than $6\mu m \times 6\mu m$. This will allow future designs to incorporate pixel level signal processing in addition to A/D conversion. In fact the multiplexed pixel block can already perform simple pixel level processing such as averaging, and spatial differencing because of the multiplexed circuit structure.

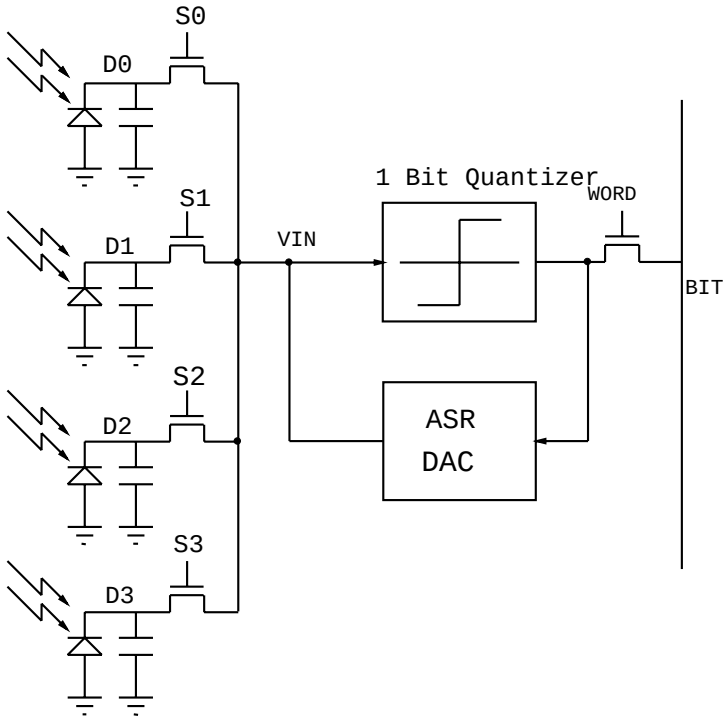


Figure 6.7: Multiplexed Pixel Block Diagram

Chapter 7

Decimation Filtering of Still Image Data

7.1 Introduction

In Chapters 4, 5, and 6 a pixel level A/D conversion technique based on sigma delta modulation was described. Although sigma delta modulation is a form of A/D conversion it does not produce a standard base two representation of the analog pixel data. Therefore, we must perform some computation on the sigma delta modulated data in order to convert it into a base two representation. This conversion is called decimation filtering. Decimation converts L one bit samples into one n bit sample, and filtering is used to reject quantization noise. Decimation filtering can be performed using various methods including nonlinear filtering and linear FIR and IIR filtering.

In this chapter we describe three decimation filtering techniques for one-bit first-order sigma delta modulated data. The first is the optimal look up table approach [36], the second is a nonlinear filtering technique [36], and the final is linear FIR filtering. All of these techniques assume that the input to the modulator is constant or equivalently the sensor is being used to capture still images. We also compare the various decimation techniques based on both computational complexity and average SNR.

Still imaging applications typically require frame rates of less than one per minute. Therefore, the average sensor bandwidth and the average computational requirement of the decimation filter will be low. This allows us to investigate decimation techniques that would not be computationally practical at higher frame rates. Implementation of the decimation filter is also simplified, because it can be operated serially at low frequencies.

7.2 Decimation Filtering Methods

There exists a large body of work on the design and analysis of decimation filters [36, 13, 59, 12]. Most of this work is based on the assumption that the input to the sigma delta modulator is a time varying bandlimited signal. In contrast, this section presents three decimation filtering techniques that assume a constant input during the sampling period. We will assume an oversampling ratio of L, i.e. each input x will be represented by L bits. All of the decimation filtering techniques discussed in this section are based on the discrete time nonlinear difference equation presented in Chapter 2 rather than a linearized approximation of the modulator. We will also assume that the sigma delta modulator is reset to a known state at the begin of each sampling period.

7.2.1 Optimal Lookup Table Decimation

Lookup table decimation transforms an L bit sigma delta modulated data sequence into a single multi-bit estimate $\hat{x}$ of the modulator input x. This transformation is onto and one-to-one and can be done using a random access memory. The address is generated by the sigma delta modulated data and the contents of the associated memory location is $\hat{x}$. $\hat{x}$ is the best estimate of the sigma delta modulators analog input x, where best is defined by minimum distance in a L2-norm, i.e. $\min_{\hat{x}} |x - \hat{x}|^2$. Although this decimation technique is impractical because it is intolerant to noise and circuit variation, as discussed in the following paragraphs, it will allow us to bound the maximum achievable SNR and investigate the nonuniform quantization

properties of one-bit first-order sigma delta modulators.

To determine the contents of the lookup table we must first find the exact location and size of each quantization interval as a function of the oversampling ratio L. A quantization interval is a range of input values such that the output code generated by the sigma delta modulator does not change. In a typical uniform quantizer each of the input intervals is a constant size, but this is not true of a sigma delta modulator. In order to find all of the quantization intervals we must find all of the transition points, or quantization interval separation points.

Proposition 3 *All of the transition points of a one-bit first-order sigma delta modulator with an oversampling ratio L can be determined using the following equation.*

$$x = \frac{pb - (2jb + u_0)}{p}, \quad 0 \leq j \leq L-1, \tag{7.1}$$

where x is a transition point, u_0 is the initial state of the integrator and is assumed to be bounded by $\pm 2b$, b is the magnitude of the one bit quantizer output, and j and p are integers with the following restrictions:

$$0 \leq j < p, \quad 1 \leq p \leq L-1 \quad u_0 > 0, \tag{7.2}$$

$$1 \leq j < p, \quad 1 \leq p \leq L-1 \quad u_0 = 0, \tag{7.3}$$

$$1 \leq j \leq p, \quad 1 \leq p \leq L-1 \quad u_0 < 0. \tag{7.4}$$

Proof:

In order to find all of the transition points we start by finding the position of $j+1$th negative output bit from the sigma delta modulator n_{j+1}. It is known that (see Chapter 2)

$$x - b \leq u_n \leq x + b, \tag{7.5}$$

and therefore

$$x - b \leq u_0 + 2jb - n_{j+1}(b - x) \leq x + b. \tag{7.6}$$

Since we assumed that $q(u_{j+1})$ is negative this implies that $u_n < 0$ and

$$x - b \leq u_0 + 2jb - n_{j+1}(b - x) < 0, \tag{7.7}$$

and thus

$$(n_{j+1} - 1)(b - x) \leq u_0 + 2jb < n_{j+1}(b - x). \tag{7.8}$$

Using these two inequalities implies that

$$n_{j+1} = \lceil \frac{u_0 + 2jb}{b - x} \rceil, \qquad 0 \leq j \leq L - 1. \tag{7.9}$$

Where $\lceil a \rceil$ indicates the ceiling function of a. We know that a transition point must occur at each change in the output code stream and therefore a transition occurs when x is such that the above equation is equal without the ceiling function, i.e.

$$\frac{u_0 + 2jb}{b - x} = p \tag{7.10}$$

and p is an integer. Now solving for x we find

$$x = \frac{pb - (2jb + u_0)}{p}, \qquad 1 \leq p \leq L - 1. \tag{7.11}$$

Now we must find the range of valid values for j. Using the assumption that $-b < x < b$ we find that

$$0 \leq j < p, \quad 1 \leq p \leq L - 1 \quad u_0 > 0, \tag{7.12}$$

$$1 \leq j < p, \quad 1 \leq p \leq L - 1 \quad u_0 = 0, \tag{7.13}$$

$$1 \leq j \leq p, \quad 1 \leq p \leq L - 1 \quad u_0 < 0 \quad \square. \tag{7.14}$$

Now that we have a method of finding the transition points we can determine all possible transition points and all of the corresponding quantization intervals. This allows us to determine the values in the lookup table and inspect the nonuniform quantization of a first order sigma delta modulator. Each value in the lookup table, $\hat{x}$, is determined by both the known quantization intervals and the input distribution

of x. These values are selected such that the average mean squared error between the analog input, x, and the lookup table value $\hat{x}$ is minimized. For example, assuming that the input to the modulator is a continuous valued random variable X, then the value $\hat{x_i}$ assigned to each quantization interval I_i is selected such that

$$\min_{\hat{\mathbf{x}}} \sum_{i=0}^{N-1} E[(X - \hat{x_i})^2 | X \in I_i] Pr(X \in I_i) \tag{7.15}$$

Where N is the number of quantization intervals, $Pr(A)$ is the probability of A, and I_i is the i-th quantization interval defined from most negative to most positive. The number of locations in the lookup table is determined by the number of quantization intervals, and each location must have at least $\lceil 10 \log_{10}(SNR) \rceil$ bits. The maximum number of quantization intervals is always less than or equal to the number of transition points plus one, and the number of transition points is determined by the number of permitted j, p sets. If we assume that $u_0 \neq 0$ then the number of quantization intervals can be counted using the following equation

$$1 + \sum_{p=1}^{N-1} p = \frac{1}{2} N(N-1) + 1. \tag{7.16}$$

For example, if $L = 1024$ the lookup table would require a 523777×19 bit random access memory. However, we have assumed that our system is both noise free and without any nonlinearities except the one bit quantizer. Note that if u_0 is allowed to vary around an assumed mean value both transition points and code words can be altered from their expected values. This will severely degrade the achievable SNR of this decimation technique. If we take these variations into account the size of the memory will grow rapidly. In fact if all of the possible imperfections caused by a standard analog circuit are taken into account the random access memory in the previous example would require $2^{1024} \times 19$ bits!

7.2.2 Nonlinear Decimation

The nonlinear decimation technique presented in this section is based on the "zoomer" algorithm [36]. This algorithm exploits the inherent structure of the sigma delta modulator output sequence. The basic idea is to calculate a series of upper and lower bound on x. Then x is estimated by taking the midpoint between the bounds. In order to determine the upper and lower bounds on x the nonlinear difference equation

$$u_{n+1} = u_n + x_n - q(u_n) \tag{7.17}$$

is rearranged, as before, into the form:

$$\frac{1}{L}\sum_{i=0}^{L-1} q(u_i) = x + \frac{u_L - u_0}{L}. \tag{7.18}$$

Since we know the value of u_0 and we know the sign of u_L we can find, for each value of L, either an upper or lower bound on x. This is shown in the following equation.

$$\frac{1}{L}\sum_{i=0}^{L-1} q(u_i) = x + \frac{u_{L-1} - u_0}{L} \geq x \quad \text{if } u_{L-1} \geq 0 \tag{7.19}$$

$$\frac{1}{L}\sum_{i=0}^{L-1} q(u_i) = x + \frac{u_{L-1} - u_0}{L} < x \quad \text{if } u_{L-1} < 0 \tag{7.20}$$

$$\tag{7.21}$$

For example, if we assume that $u_0 = 0$ and $q(u_L) = b$ we know that $\frac{1}{L}\sum_{i=0}^{L} q(u_i)$ is a lower bound on x, or if $u_0 = 0$ and $q(u_L) = -b$ we know that $\frac{1}{L}\sum_{i=0}^{L-1} q(u_i)$ is an upper bound on x. The zoomer algorithm uses this observation to determine the best upper and lower bounds over all of the L input bits. Then x is estimated using the following equation.

$$\hat{x} = \frac{\text{upper bound} - \text{lower bound}}{2} \tag{7.22}$$

Pseudo code for the zoomer algorithm is shown in Figure 7.1. The main drawback of this algorithm is its sensitivity to errors in the initial condition. For example, an initial condition error of $0.024b$ when u_0 should be equal to zero reduces the average

```
ZoomerAlgorithm(SigmaDeltaModulatorOutputVector,xhat)
   minx = -b;
   maxx = b;
   sum = SigmaDeltaModulatorOutputVector(0);
   i = 1;
   While (i < L) {
      if (SigmaDeltaModulatorOutputVector(i) >= 0) {
         if (minx < sum/i) {
            minx = sum/i;
         }
      }
      else {
         if (maxx > sum/i) {
            maxx = sum/i;
         }
      }
      sum = sum + SigmaDeltaModulatorOutputVector(i);
      i = i + 1;
   }
   xhat = (maxx+minx)/2;
```

Figure 7.1: Zoomer Algorithm

output SNR of the decoder by 2dB, and further increases in error cause very rapid degradation of SNR.

If the initial condition u_0 of the sigma delta modulator is unknown but bounded we can use the LP algorithm. The LP algorithm is a direct extension of the zoomer algorithm. Instead of just bounding x the LP algorithm bounds both x and u_0 and then uses this information to estimate the input of the modulator x. This is again done by looking at the nonlinear difference equation and rearranging it as we did in the zoomer algorithm

$$\frac{1}{L}\sum_{i=0}^{L-1} q(u_i) = x + \frac{u_L - u_0}{L}. \tag{7.23}$$

Now for each value of L we have a linear inequality.

$$\frac{1}{L}\sum_{i=0}^{L-1} q(u_i) \geq x - \frac{u_0}{L} \quad \text{if } u_L \geq 0 \tag{7.24}$$

$$\frac{1}{L}\sum_{i=0}^{L-1} q(u_i) < x - \frac{u_0}{L} \quad \text{if } u_L < 0 \tag{7.25}$$

$$\tag{7.26}$$

If it is assumed that u_0 is bounded by $-K$ and K and K is a positive real number $K < b$, the above equations form $L+1$ linear inequalities. These inequalities can be arranged in the following format.

$$\mathbf{ab} \leq \mathbf{c}, \tag{7.27}$$

where $\mathbf{a}$ is a matrix of size $2 \times L+1$ containing the coefficients of x and u_0, $\mathbf{b}$ is a vector of size 1×2 containing x and u_0, and $\mathbf{c}$ is vector of size $L+1$ containing the sum $\frac{1}{L}\sum_{i=0}^{L-1} q(u_i)$. Using standard linear programming techniques [4] we find

$$x_{max} = \max_x(\mathbf{ab} \leq \mathbf{c}), \tag{7.28}$$

$$x_{min} = \min_x(\mathbf{ab} \leq \mathbf{c}), \tag{7.29}$$

and estimate x by

$$\hat{x} = \frac{x_{max} + x_{min}}{2}. \tag{7.30}$$

This method produces very good results if the known bound on u_0 is small. For example, if the the initial condition is bounded between $-b/8$ and $b/8$ the SNR is within 0.5dB of the case when u_0 is known exactly. In contrast, if the bound on u_0 is rather poor, i.e. $-2b < u_0 < 2b$ then the decimated results are 5-10dB lower than when u_0 is known exactly. If u_0 is known exactly the LP algorithm becomes the zoomer algorithm. Note that both the zoomer and LP algorithms suffer from SNR degradation caused by integrator leakage. We will not discuss this problem here, but [36] presents some results in this area.

7.2.3 FIR Decimation

Finite impulse response decimation uses a weighted linear combination of L output samples from a sigma delta modulator to produce an estimate, $\hat{x}$, of the modulators input x, i.e.

$$\hat{x} = \sum_{i=0}^{L-1} d_i q(u_i). \tag{7.31}$$

Several different families of coefficient values d_i have been studied including uniform weighting where $d_i = 1/L$ and triangular weighting [33]. Neither of these techniques produces the optimal FIR decimator. The optimal FIR decimator cannot be derived in closed form, but a very close approximation can be achieved by using Monte Carlo simulation and standard optimization techniques. The basic problem can be formulated as follows: find a set of coefficients d_i, $\mathbf{d}$, such that

$$\min_{\mathbf{d}} E_{X,U_0}[(\sum_{i=0}^{L-1} d_i q(u_i) - X)^2]. \tag{7.32}$$

Unfortunately $q(u_i)$ is a discontinuous function of x and therefore standard nonlinear optimization techniques based on continuous convex functions are not applicable [37]. Therefore, other techniques must be used to find an approximate minimum. We used a Monte Carlo simulation followed by a simple quadratic optimization. The Monte Carlo simulation is used to produce values of $q(u_i)$. X and U_0 are random variables selected at the beginning of each simulation. The simulation is run M times and the data is arranged in a $L \times M$ matrix, $\mathbf{r}$. Then by finding a solution to the following

optimization problem,

$$\min_{\mathbf{d}}(\mathbf{rd}-\mathbf{x})^T(\mathbf{rd}-\mathbf{x}) \tag{7.33}$$

we determine the approximately optimal vector $\mathbf{d}$. $\mathbf{x}$ is the vector of input values used in the Monte Carlo simulation to create the $\mathbf{r}$ matrix. The solution to the above optimization problem has the following closed from solution.

$$\mathbf{d} = (\mathbf{r}^T\mathbf{r})^{-1}\mathbf{x} \tag{7.34}$$

Note that this technique for finding an approximately optimal solution is not dependent on the assumed input distribution and initial condition distribution. The only assumption necessary is that the modulator does not overload, i.e. $-b < x < b$.

7.3 Simulation and Analysis of Decimation Filtering Techniques

In order to compare the decimation filtering techniques we will use two separate metrics. The first is average SNR as a function of oversampling ratio L, and the second is the computational complexity of the technique. SNR refers to the average signal power divided by the average noise power. Noise power is the average mean squared error or L2-norm between the decimated output and the analog input of the sigma delta modulator.

7.3.1 Optimal Lookup Table Decimation

The average SNR of the optimal lookup table decimation filter was calculated by assuming that the input to the modulator X is uniformly distributed between $-0.8b$ and $0.8b$ and that the initial condition of the modulator $u_0 = \frac{b}{\pi}$. The average noise was calculated by

$$\text{Noise Power} = \sum_i E[(X-\hat{x_i})^2|X \in I_i]Pr(X \in I_i). \tag{7.35}$$

Note that if the input X is uniformly distributed the optimal[1] $\hat{x}_i$ is just the center point of its quantization interval with the exception of the end points. The two end points were selected as follows, the most negative $\hat{x}_i$ was selected as the center point between $-0.8b$ and the first transition point above $-0.8b$. The most positive $\hat{x}_i$ was selected as the center point between $0.8b$ and the first transition point before $0.8b$. The signal power was calculated by

$$\text{Signal Power} = \sum_i (\hat{x}_i{}^2 Pr(X \in I_i)) \tag{7.36}$$

The average SNR was calculated by

$$\text{SNR} = 10 \log_{10}(\frac{\text{Signal Power}}{\text{Noise Power}}) \tag{7.37}$$

Figure 7.2 shows the average SNR for the optimal lookup table decimation filter as a function of oversampling ratio L. These results are an upper bound on the achievable average SNR for any decimation filter.

The computational complexity of this algorithm is very low. In fact, since of the decimation process is accomplished with a single memory access in a lookup table it is O(1).

7.3.2 Nonlinear Decimation

The average SNR versus oversampling ratio for both the zoomer and LP algorithms were measured using Monte Carlo simulation. Each Monte Carlo simulation was performed using L^2 iterations, where L is the oversampling ratio of the sigma delta modulator. L was logarithmically varied between 8 and 64. Each iteration of the Monte Carlo simulation was performed by randomly selecting an input value x from a given distribution and selecting an initial condition u_0 and simulating L samples from a one-bit first-order sigma delta modulator. Then, this sigma delta modulated data was decimated using either the zoomer or LP algorithm. In order to test the

[1] Optimal in the sense of minimum mean square error between x and $\hat{x}$.

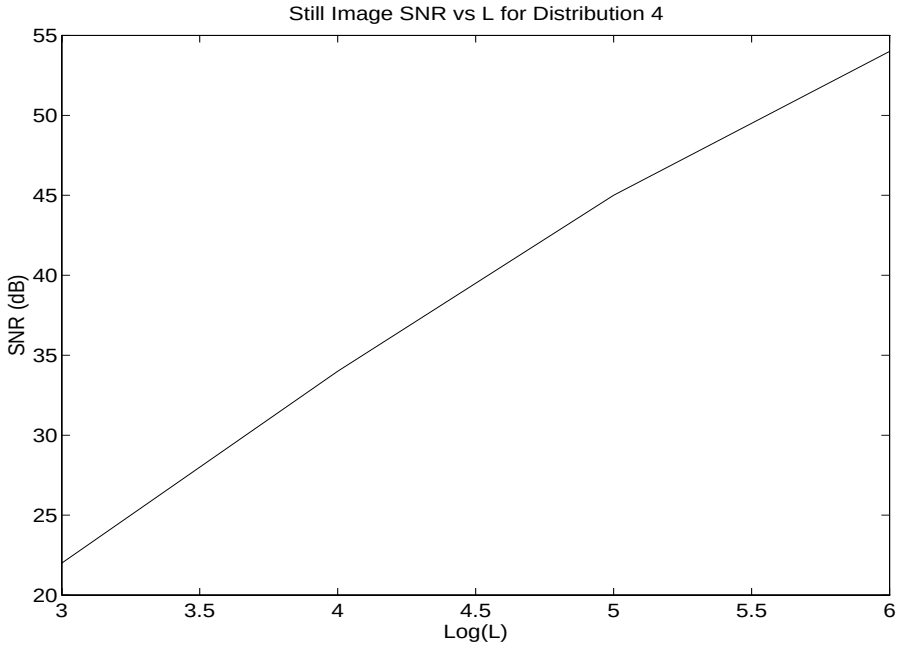


Figure 7.2: SNR of lookup table decimation filter as a function of oversampling ratio L.

zoomer and LP algorithms sensitivity to data, four different input distributions were used for generating the sigma delta modulated data. The histograms of the four distributions are shown in Figure 7.3. The first two distributions were derived from a set of 150 still images, and the last two are just uniform distributions with different widths. The zoomer algorithm uses an initial condition $u_0 = 0$, and the LP algorithm uses two separate distributions for u_0. The first distribution is uniform between $\pm 0.1b$, and the second is uniform between $\pm 2b$. Figures 7.4, 7.5, 7.6, and 7.7 show the results of the various simulations.

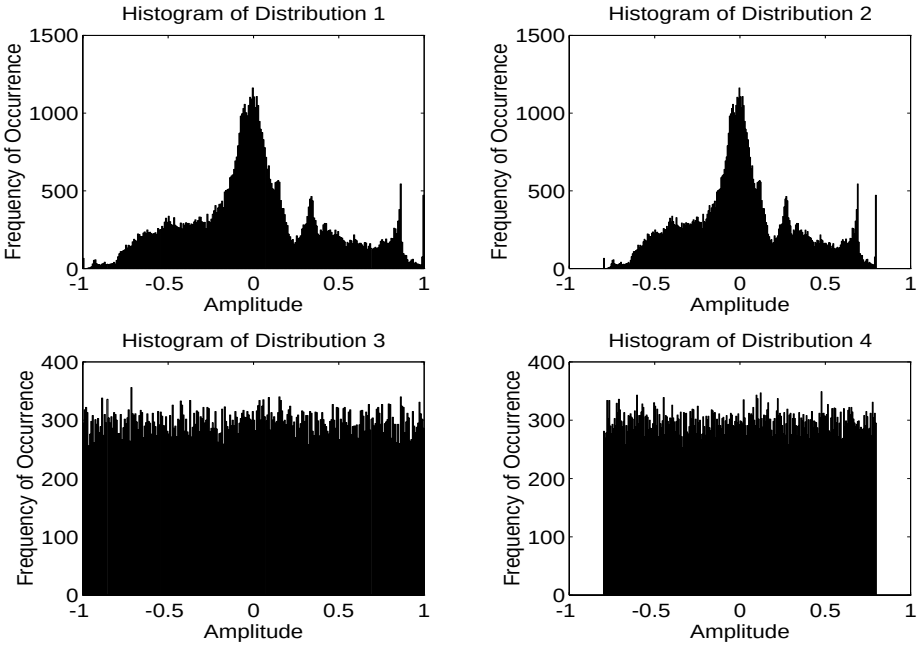


Figure 7.3:

These results show that if the initial condition of the modulator is known, a higher SNR can be achieved than if the initial condition is unknown. Although this result is not surprising, it is interesting to note that if the initial condition is unknown but bounded within $\pm 0.1b$ the LP algorithm can achieve results within 0.5dB of the zoomer algorithm using a known initial condition. The modulator's input distribution also effects the SNR by up to 8dB. Also note that reducing the modulators input to $\pm 0.8b$ increases the average SNR by 2-3dB.

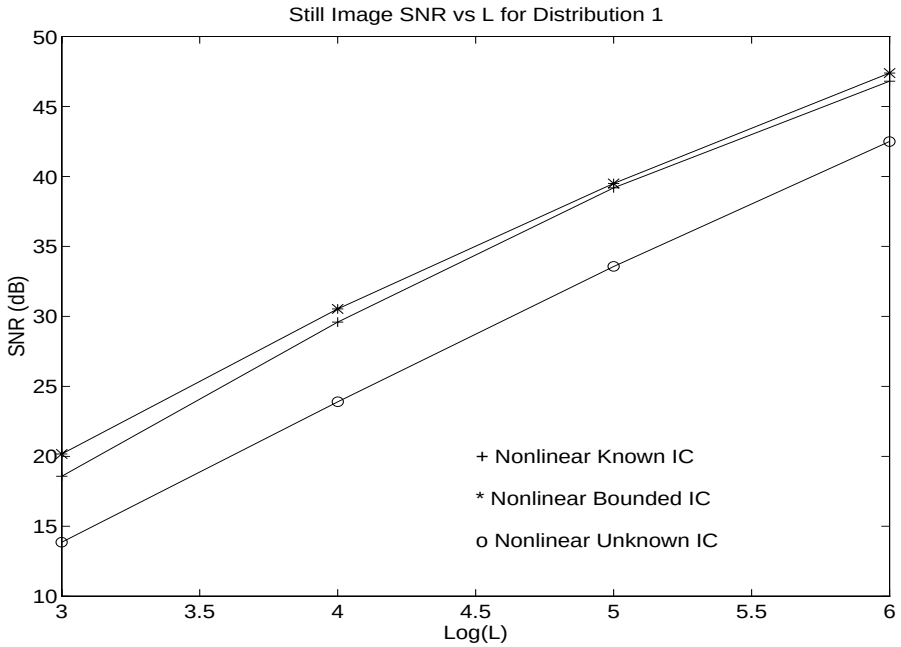


Figure 7.4: Distribution1: SNR of nonlinear algorithms verses oversampling ratio L.

The computational complexity of the zoomer algorithm is $O(L)$. This can be seen by noting that the algorithm only inspects each bit of the sigma delta modulated data sequence once, and performs a constant number of operations per bit. On the other hand the LP algorithm uses linear programming. If the linear programming problem is solved using the Karmarkar algorithm the LP algorithm has a computation complexity of $O(L^{4.5})$ [4]. The added computational complexity of the LP algorithm is offset by its robust operation in the face of unknown initial conditions.

7.3.3 FIR Decimation

The same Monte Carlo simulation method described in the last subsection was also used to measure the average SNR of three different FIR filter techniques. These techniques include the filter we proposed in the previous section, a uniformly weighted filter, and a triangularly weighted filter. We also used the same four modulator input distributions described in the last subsection, but u_0 is assumed to be uniformly

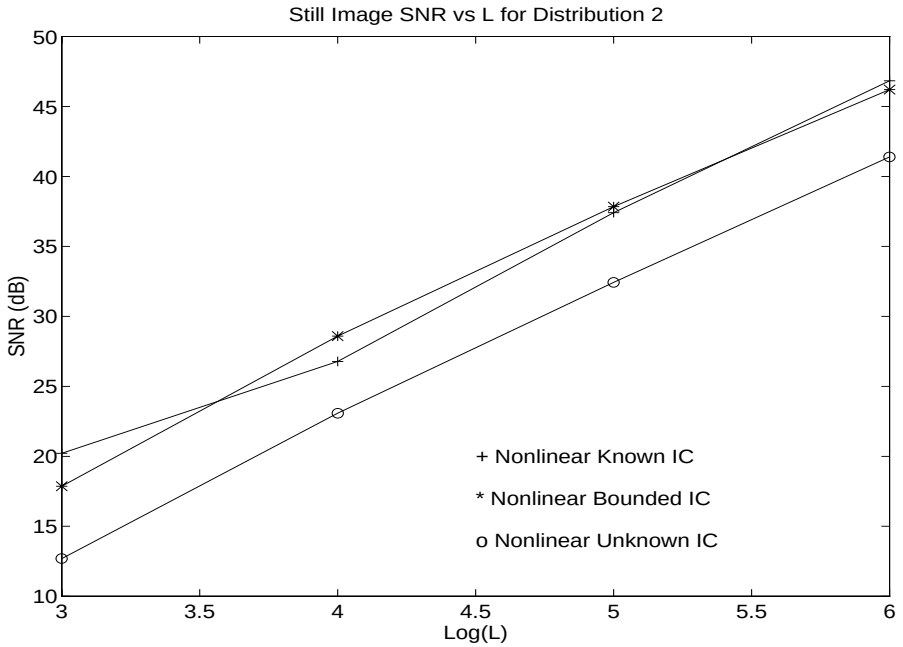


Figure 7.5: Distribution2: SNR of nonlinear algorithms verses oversampling ratio L.

distributed between $-2b$ and $2b$. Figures 7.8, 7.9, 7.10, and 7.11 show results of the various simulations. Each figure was generated using a different modulator input distribution.

These results show that the uniformly weighted filter achieves an SNR increase of 6dB/octave, while both of the other techniques achieve 9dB/octave. It can also be seen that the approximately optimal filter achieves an SNR gain of 1dB over the triangularly weighted filter. The modulator input distributions affect the average SNR in exactly the same way as noted in the last subsection. This implies that the average SNR as a function of modulator input distribution is independent of the decimation technique.

The computational complexity of any FIR filter technique is $O(L)$. This is true because FIR filter techniques are performed using only a single dot product between a set of filter coefficients and the sigma delta modulated data.

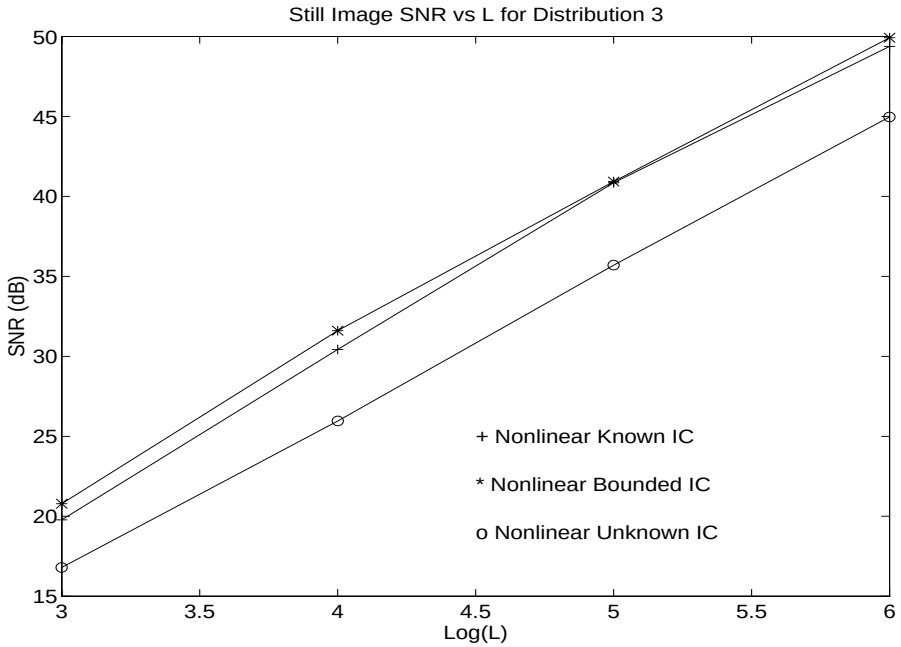


Figure 7.6: Distribution3: SNR of nonlinear algorithms verses oversampling ratio L.

7.4 Discussion

Figure 7.12 shows a comparison of all the decimation techniques in the previous section. We assume that X is uniformly distributed between $\pm 0.8b$, and the initial condition varies from technique to technique based on the assumptions in the previous section. The lookup table and zoomer techniques provide the highest SNR but assume an almost perfect sigma delta modulator model. In fact, for most applications they are impractical because their performance quickly degrades in the face of non ideal system parameters such as unknown initial conditions. The LP algorithm overcomes the initial condition limitation imposed by the lookup table and zoomer techniques, but in the case where the initial condition is unbounded, i.e. $-2b < u_0 < 2b$, it does not exceed the SNR achieved by FIR techniques. Therefore, we believe that the LP algorithm provides the best overall performance if the initial condition can be bounded within a small region, but if it can not then our FIR technique provides the best overall performance.

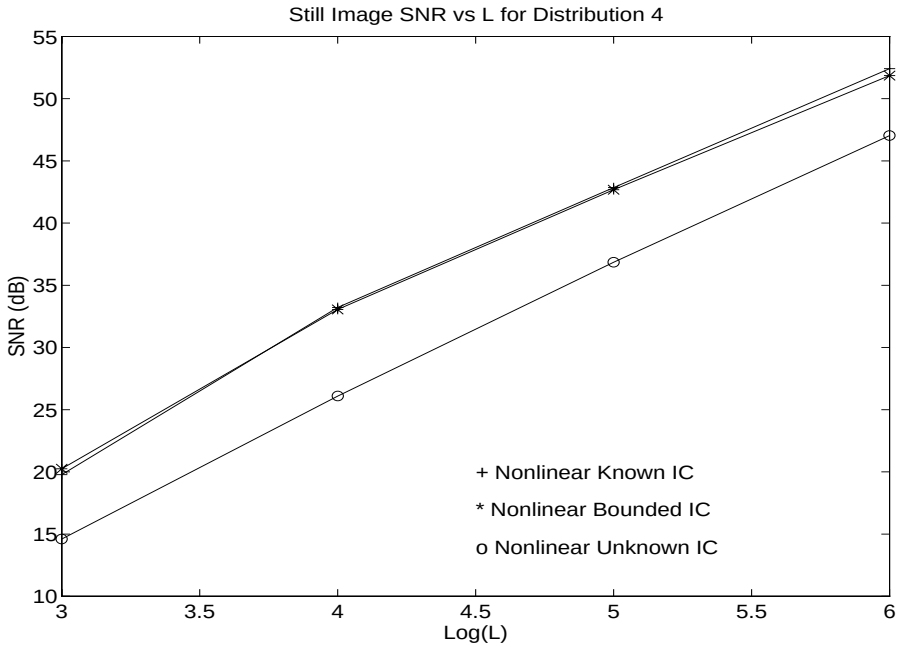


Figure 7.7: Distribution4: SNR of nonlinear algorithms verses oversampling ratio L.

7.5 Summary

In this chapter we have investigated three separate types of decimation filters. All of these filters are intended to decimate still image data generated by our sensor. The first was a lookup table approach which achieved the highest SNR for a given oversampling ratio, but its utility is limited by its sensitivity to non ideal conditions. The second was a nonlinear technique based on the zoomer algorithm. This technique also achieved a high SNR, but again, suffers from sensitivity to unknown initial conditions. The LP algorithm extends the zoomer algorithm to the case where the modulators initial condition is unknown. This technique can achieve high SNR, within 0.5dB of the maximum achievable, if the modulators input distribution can be bounded within a small range such as $\pm 0.1b$. The final technique was an FIR filter which achieved the lowest SNR. However, it is robust to initial conditions. If the initial condition is unbounded, i.e. $-2b < u_0 < 2b$, it does as well as the LP algorithm.

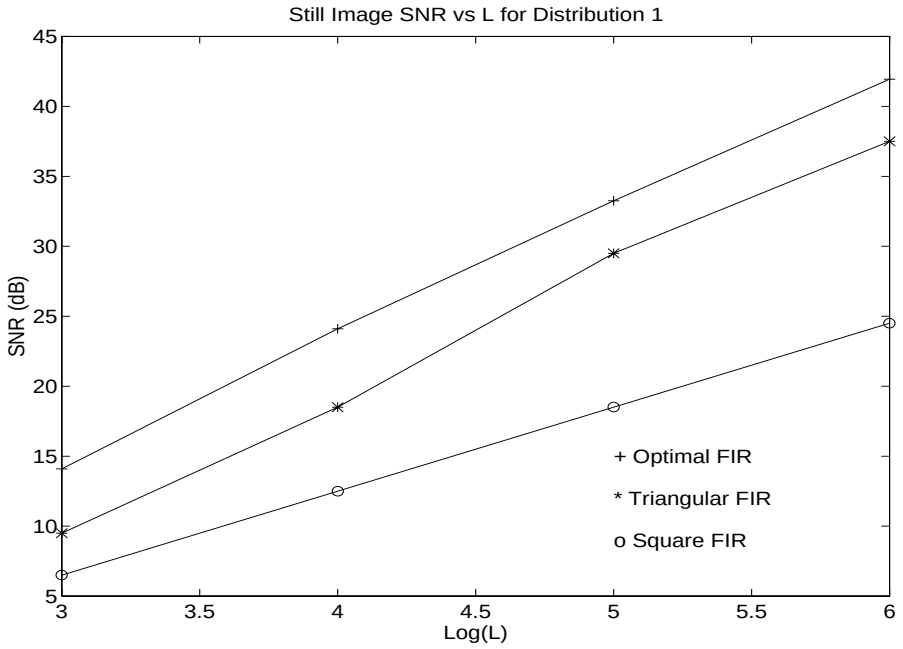


Figure 7.8: Distribution1: SNR of FIR filters verses oversampling ratio L.

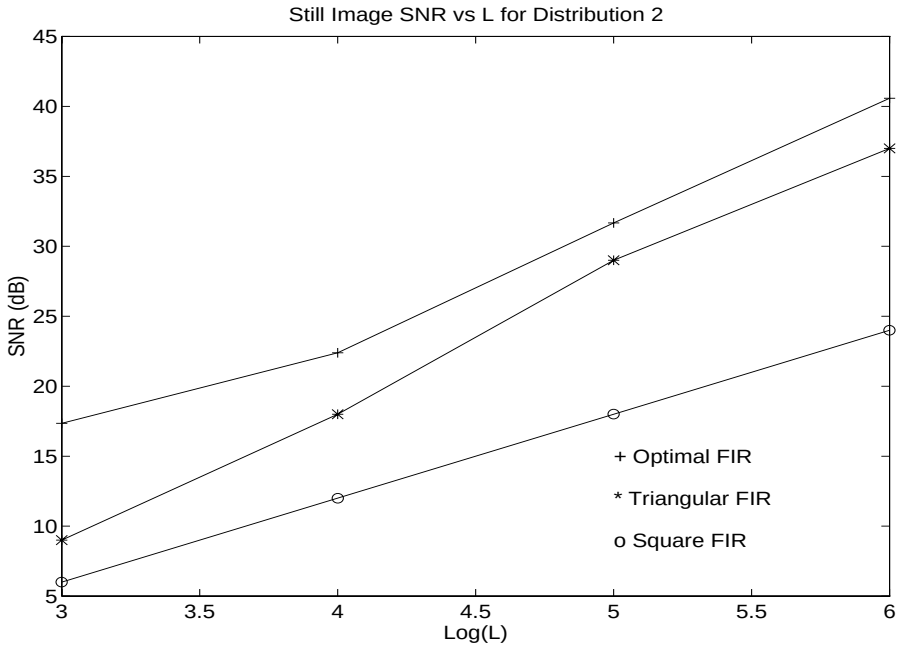


Figure 7.9: Distribution2: SNR of FIR filters verses oversampling ratio L.

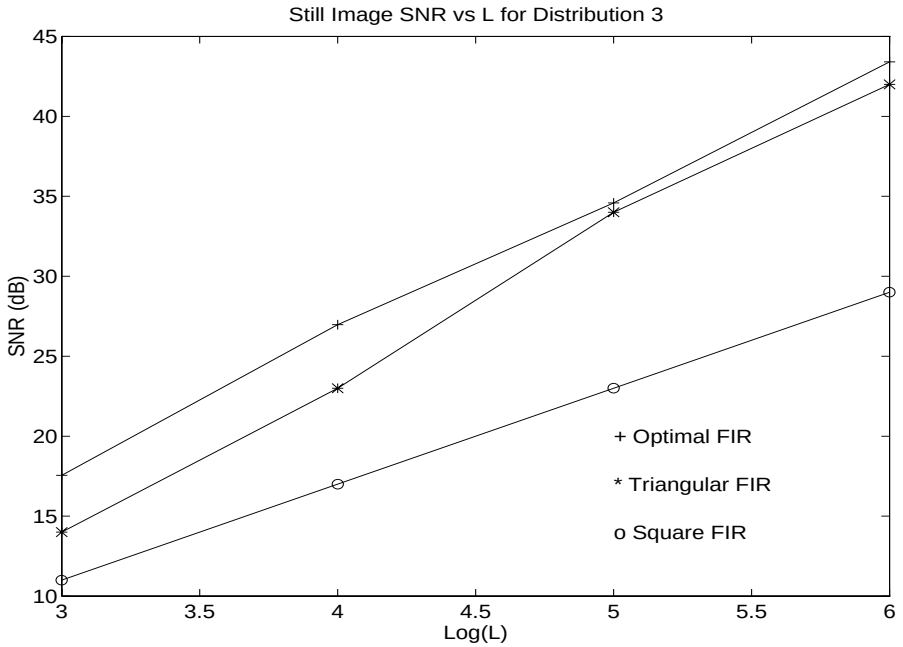


Figure 7.10: Distribution3: SNR of FIR filters verses oversampling ratio L.

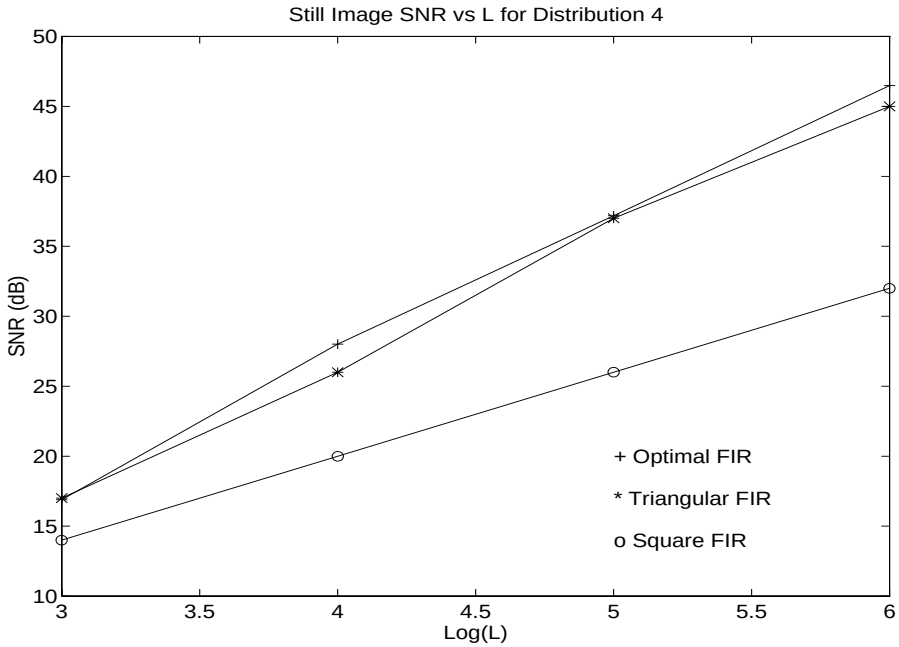


Figure 7.11: Distribution4: SNR of FIR filters verses oversampling ratio L.

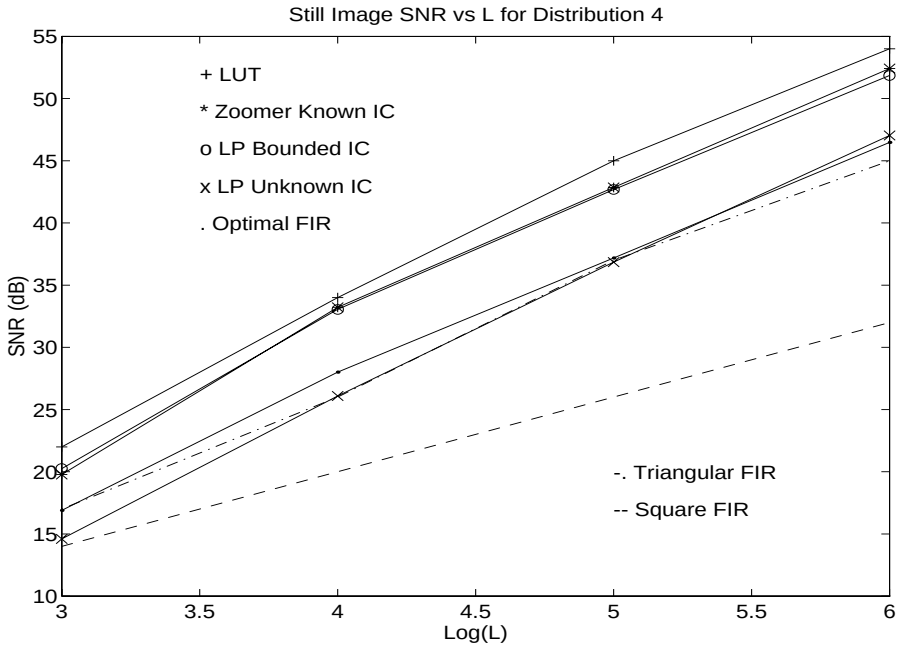


Figure 7.12: A comparison of all of the decimation techniques as a function of L.

Chapter 8

Decimation Filtering of Video Data

8.1 Introduction

In Chapter 7 we discussed several techniques for decimating still image data. Our main goal in that chapter was to determine which of the various techniques achieved the highest SNR for a given oversampling ratio. This was reasonable because we assumed that our sensor produced very low data rates while capturing still images. In this chapter we extend this work to video data. This task is quite daunting because of the high data rates generated by our image sensor. For example, a 1000×1000 pixel image sensor operating at 30 frame per second with an oversampling ratio of 64 produces 1.92 billion bits per second. All of this data must be decimated in real time and all of the decimation circuitry must be on the same chip as the image sensor. Therefore, the decimation techniques we will discuss must take into account circuit size, power dissipation, memory size, speed, and average SNR.

This chapter begins with a discussion of the system level considerations necessary for implementing a real time decimation circuit on the same chip with our image sensor. This includes the time versus amplitude resolution trade-offs associated with converting sigma delta modulated video data into standard video frames. Then several different linear and nonlinear decimation filtering techniques are discussed. This discussion focuses on both performance and implementation issues. Only one of the decimation techniques discussed, single stage FIR, was determined to be acceptable

for our application in terms of circuit and memory size. Three different single stage FIR filters are compared based on average SNR. An implementation example of this technique is also presented.

In this chapter we assume that the input to each sigma delta modulator is a time varying signal and that the modulator state is never reset.

8.2 System Level Considerations for Video Decimation

As discussed in Chapter 4, a pixel level sigma delta modulated image sensor is different from the typical integration sampled image sensor in terms of its frequency and amplitude resolution. A sigma delta modulated image sensor can trade-off frequency and amplitude resolution during the decimation processes. Each pixel of our sigma delta modulated image sensor produces a continuous stream of bits. These bits can be decimated at different rates in order to take advantage of the trade-offs associated with the time-amplitude resolution of a sigma delta modulator. If the images focused on the sensor are moving quickly then the effective oversampling ratio can be reduced in order to achieve higher temporal resolution at the cost of lower amplitude resolution. On the other hand, if the images are almost still the effective oversampling ratio can be increased reducing the temporal resolution while increasing the amplitude resolution. For example, if the sensor is being clocked at 2kHz and we want to observe images moving at 60Hz we would decimate the data at an oversampling ratio of 16. This would produce a 120Hz frame rate from our sensor with an average SNR per pixel of approximately 30dB. If we wanted to observe images moving at 1Hz we would decimate the data at an oversampling ratio of 1000. This would produce a 2Hz frame rate from our sensor with a theoretical average SNR per pixel of approximately 84dB!

After sampling rate and the oversampling ratio are determined there is still one free parameter in the decimation process. This parameter is the size of the decimation window, i.e. the number of sigma delta modulated bits used to produce each pixel

value. Figure 8.1 shows a graphical representation of a decimation window of length $2L$, where L is the oversampling ratio. Unlike decimation of still image data, the decimation window is not limited to the oversampling ratio. In fact the decimation window could be the length of the entire data sequence. Although this is impractical, the idea will prove to be useful in determining an upper bound on the achievable SNR for any FIR decimation filter. Wide decimation windows increase the achievable SNR because they allow us to exploit redundancy in the data, bit wide windows also increase the external memory and the required speed of the decimation filters. This is discussed in the following sections. Therefore, there is a trade-off between achievable SNR and required system resources, i.e. external memory and I/O bandwidth.

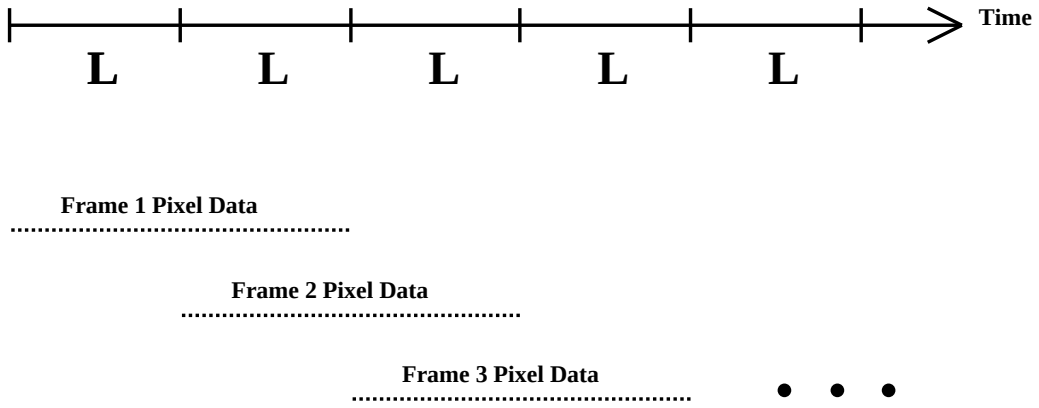


Figure 8.1: Decimation Window

Figure 4.5 shows a design of an integrated video decimation circuit. It consists of three separate pieces. These include our CMOS area image sensor with pixel level A/D conversion, a decimation filter, and external memory for storage of the video data. The exact size of the external memory and the method in which it interfaces with the image sensor and the decimation filter (or filters) depends on the decimation technique. These details will be further discussed in the next section when the various decimation techniques are presented.

8.3 Decimation Filter Techniques

In this section we present both nonlinear and linear decimation techniques.

8.3.1 Nonlinear Decimation Filters

In the last chapter we showed that nonlinear decimation techniques achieve higher average SNR than linear techniques, but they also require larger amounts of computational effort. The promise of high SNR led us to investigate two nonlinear decimation techniques [36, 66] for video data. Both of these techniques are based on the theory of Projections Onto Convex Sets (POCS) [49]. POCS is an iterative algorithm that alternates between time-domain and frequency-domain projections in an attempt to find a signal that is invariant under both. We will concentrate on the POCS based algorithm developed by Hein and Zankhor [36].

In order to properly describe this algorithm a bit of notation must be developed. For any binary output signal $\mathbf{y} = \{y_0, \cdots, y_{L-1}\}$ from a sigma delta modulator, we define the set S_1 of all input signals $\mathbf{x} = \{x_0, \cdots, x_{L-1}\}$ that results in $\mathbf{y}$ when applied to the sigma delta modulator. We also define the set S_2 of all L-sample signals $\mathbf{x}$ that are bandlimited. Where bandlimited is strictly defined in [36]. To estimate the input signal we must find $\hat{\mathbf{x}} \in S_1 \cap S_2$. S_1 and S_2 are shown to be convex [1] in [36] as assumed by the POCS algorithm. If we denote the orthogonal projections on S_1 and S_2 by P_1 and P_2, respectively, then the theory of POCS states than an element $\hat{\mathbf{x}} \in S_1 \cap S_2$ can be found from any initial guess $\mathbf{x}_0$ by the iteration

$$\mathbf{x}_{n+1} = (P_1 \circ P_2)\mathbf{x}_n, \qquad n \geq 0. \tag{8.1}$$

The POCS theory also state that

$$\hat{\mathbf{x}} = P_1\hat{\mathbf{x}} = P_2\hat{\mathbf{x}} = \lim_{n\to\infty} \mathbf{x}_n. \tag{8.2}$$

Results presented in [36] show that this algorithms average SNR is 20-30dB higher

[1] A set S is convex if for all a, $b \in S$, $\alpha a + (1-\alpha)b \in S$ for any $0 < \alpha < 1$.

than the optimal linear low filter for oversampling ratios between 64 and 128. Although, this algorithm achieves a high average SNR it requires very high computational complexity. In order to perform the POCS iteration, and decimate the sigma delta modulated data, a quadratic programming problem [37] with at least L linear constraints and $2L$ variables must be solved. This iteration must also be performed several times for adequate convergence. Both of these claims are proved in [36]. Note that the POCS algorithm must be performed for each pixel in the sensor during each frame of data. This type of computational complexity makes this algorithm impractical for our application. Therefore, we must look for a linear decimation technique that meets our requirements.

8.3.2 Multistage/Multirate Linear Decimation Filters

The advantages of lowpass multistage/multirate linear decimation filters have been widely discussed in the literature [59, 31, 15, 14, 13]. These decimation filters are known for being able to produce very narrow transition bands and highly attenuated stop bands without a large number of filter coefficients. This is an important feature for most VLSI implementations, because it can reduce the size of the decimation filter.

Multistage decimation filters can have a variable numbers of stages depending on the application and the amount of decimation required. Most applications use two stages because of the simplicity of the design, but other applications require more stages in order to reduce the decimation filters size [59]. In this subsection we focus on two stage decimation filters. A typical two stage multirate decimation filter is shown in Figure 8.2. The first stage is used to decimate the sigma delta modulated data to a frequency above the Nyquist rate, usually around 4 times the Nyquist rate. This decimation stage usually uses a simple FIR filter, such as a $(\frac{\sin \pi x}{\pi x})^N$ discussed on page 10 of [12]. The second stage decimates the data to the Nyquist rate. This filter is typically more complicated because it must attenuate quantization noise above the Nyquist rate and correct for any passband distortion caused by the first stage. Each transfer function, $H_i(z)$, within a multistage decimation filter can by implemented

using either FIR or IIR techniques [52]. Since IIR filters require additional memory within the feedback path they are not practical for our application, because we want to minimize the amount of external memory. Therefore, we will concentrate on FIR filters for the rest of this chapter.

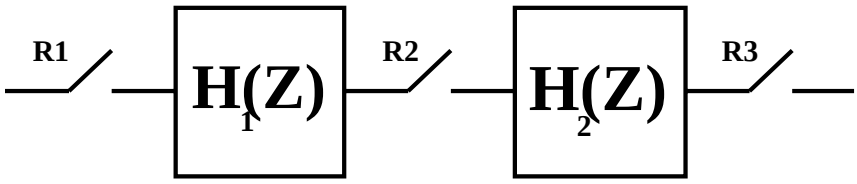


Figure 8.2: Two stage multirate decimation filter Block Diagram.

In order to illustrate how a multistage decimation filter can be integrated with our image sensor a complete example is presented and its operation is described. Figure 8.3 shows a block diagram of the sensor, decimation filters and the external memory. Assume that the image sensor has 1000×1000 pixels and that the I/O bus width is 64 bits. The number of parallel decimation filters is selected to accommodate the I/O bus width. The lower bound on the I/O bus width for each decimation filter is the desired number of bits per pixel. The pixel level sigma delta modulators are sampled at 2kHz, and the video data is decimated by a factor of 64. Resulting in a 30Hz frame rate with approximately 7-8 bits of resolution per pixel. Therefore, 8 parallel decimation filters are integrated with the sensor. Each decimation filter consists of two stages. The first stage down samples the video data to 120Hz, and the second stage down samples the output of the first filter to 30Hz. The first filter is a 16 tap FIR and the second filter is a 32 tap FIR. Using this example the system operate as follows: after each 500μs period, i.e. one clock cycle, our image sensor generates a bit plane. During the next 500μs period this data is transported from the pixels to the decimation filters, one row at a time. Since there are only 8 decimation filters and there are 1000×1000 pixels we must multiplex the sigma delta modulated data

streams between the decimation filters. Moreover, we will multiplex the decimation filters row by row and between each set of 8 columns. This will allow us to decimate the data one bit plane at a time instead of having to store all of the bit planes and then decimate them. Bit plane data is transferred row by row to the edge of the sensor, and it is multiplexed into groups of 8 bits. Each group of the 8 bits is processed by the decimation filters using the pixels state. The state of each pixel is the result of decimating the last n bits, generated by the pixel, in the current frame. After 16 bit planes the first stage of the decimation filter produces an output, and after the first stage produces 32 values the second stage produces a single output. Therefore, a total of 16×32 clock cycles are require to produce a pixel value, i.e. the effective decimation window is 512. Since the sigma delta modulated pixel data is decimated by a factor of 64 and the decimation window is 512 we must calculate 8 frames in parallel, i.e. $\frac{512}{64} = 8$.

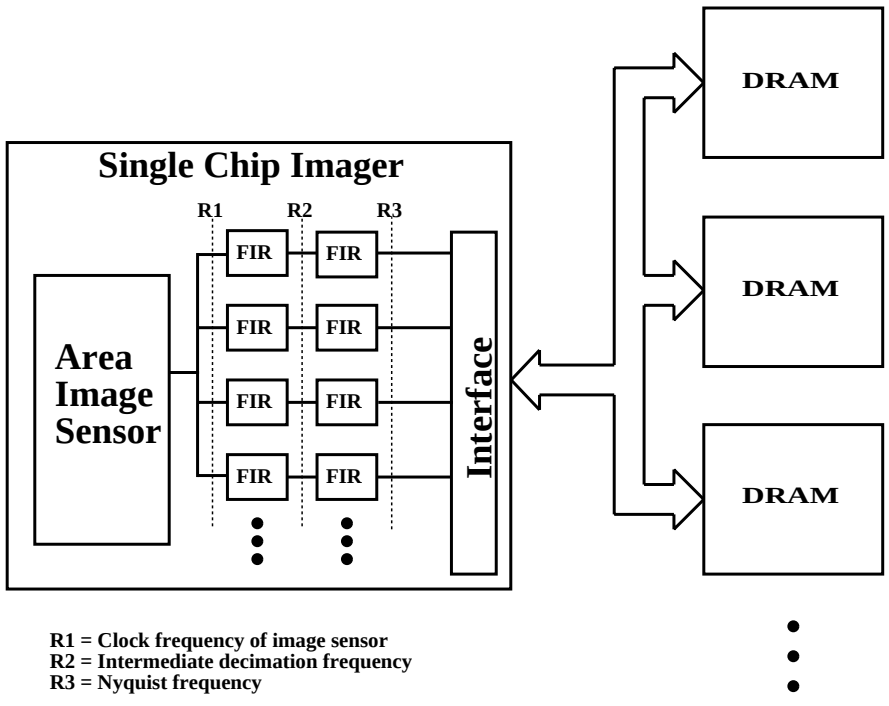


Figure 8.3: System block diagram of a multirate FIR decimation filter with our image sensor.

Since 8 frames must be calculated in parallel $8 \times 1000 \times 1000 \times 10$ bits of external

memory are required. Where 8 is the number of bits per pixel, 1000×1000 is the number of pixels, and 10 is one intermediate buffer for the first stage FIR and 8 intermediate frame buffers for the second stage FIR and one output buffer. In comparison, a typically CCD based digital camera requires a $8 \times 1000 \times 1000 \times 2$ bit frame buffer. This assumes that video or dual ported RAM is not used, i.e. we are using double buffered memory. Therefore, this design requires a factor 5 times more memory! This is not acceptable for low cost applications. The memory problem could be reduced by redesigning the filter, but the characteristic narrow transition band of multistage filters is based on the effective decimation window size. As stated previously, wider decimation windows allow narrower transition bands to be achieved, but they also increase the amount of external memory. The only advantage multistage decimation filters have over signal stage filters, in our application, is the number of filter coefficients [14]. Since these coefficients are used by all of the decimation filters, in our example 8, the total number of coefficients is relatively unimportant in terms of the entire decimation filter size. Finally, note that a single stage decimation filter only requires additions, because the inputs are just 0 or 1 but a two stage decimation filter typically requires fixed point multiplication in the second stage filter.

8.3.3 Single Stage Linear Decimation Filters

After investigating nonlinear and multistage linear decimation filters, we have determined that single stage FIR decimation filters best suit our application. They use a relatively small amount of on chip circuitry and they can be designed with minimal external memory. There are many different techniques for designing single stage FIR filters discussed in [52]. We selected to use linear estimation theory [53]. This technique uses the statistics of the sigma delta modulated data in order to produce the optimal linear estimate of the input. Optimal refers to minimum mean squared error.

Using linear estimation theory the coefficients of the FIR decimation filter are calculated as follows: assuming the same notation as Chapter 2 and that the statistics of both the input X_n and output Y_n of the sigma delta modulator are available, we

want to find a set of filter coefficients a_i such that

$$\min_{\mathbf{a}}(X_n - \sum_{i=0}^{N-1} a_i Y_{n+i-\frac{N}{2}})^2. \tag{8.3}$$

Using the orthogonality principle [53] we know that the last equation is satisfied if

$$E[(X_n - \sum_{i=0}^{N-1} a_i Y_{n+i-\frac{N}{2}})Y_m] = 0 \quad \forall \quad m \in \{n - \frac{N}{2} \cdots n + \frac{N}{2} - 1\}. \tag{8.4}$$

This simplifies to

$$E[X_n Y_m] = \sum_{i=0}^{N-1} E[a_i Y_{n+i-\frac{N}{2}} Y_m], \tag{8.5}$$

and therefore the FIR coefficient vector $\mathbf{a} = \{a_0 \cdots a_{N-1}\}$ is

$$\mathbf{a} = \mathbf{\Sigma}_{\mathbf{YY}}^{-1} \mathbf{\Sigma}_{X_n \mathbf{Y}}. \tag{8.6}$$

Where $\mathbf{\Sigma_{YY}}$ is the autocorrelation matrix of the random vector $\mathbf{Y}$, and $\mathbf{\Sigma}_{X_n \mathbf{Y}}$ is the cross correlation vector between the random variable X_n and random vector $\mathbf{Y}$.

Now that we have derived the coefficients for the FIR filter we must make some engineering decisions about the number of taps for each FIR filter, and the detains of the circuit implementation. The number of FIR taps is a trade-off between achievable SNR and the amount of external memory. As discussed in the last subsection the achievable SNR and the number of pixels sets a lower bound on the size of the external memory. The size of this memory must be increased if the width of the decimation window is larger than the oversampling ratio. In fact the size of the external memory must be at least

$$M \times N \times N \times (1 + \lceil \frac{\text{TAPS}}{L} \rceil). \tag{8.7}$$

Where M is the number of bits per pixel, $N \times N$ is the number of pixels, TAPS in the number of coefficients used by the FIR filter, and L is the oversampling ratio. The first three terms in the above equation are obvious, but the last expression needs some explanation. The 1 in $(1 + \lceil \frac{\text{TAPS}}{L} \rceil)$ refers to the output buffer, because we have assumed a double buffered output. The term $\lceil \frac{\text{TAPS}}{L} \rceil$ is the number of frames that

must be calculated in parallel. As a compromise between SNR and external memory we selected the number of taps to be $2L$ or twice the oversampling ratio. Therefore, the external memory must be at least $8 \times 1000 \times 1000 \times 3$ bits, this is only 1.5 times larger than the equivalent CCD based system. This decision also implies that two video frames must be calculated in parallel.

Figure 8.4 shows a block diagram of the single stage FIR decimation filter, with an image sensor and external memory. Figure 8.5 shows an expanded view of the parallel decimation circuitry. Each decimation filter is composed of one M bit full adder, one M bit register, and multiplexing circuitry for reading and writing the contents of the register. All of the filter coefficients are stored in one location and shared between all of the filters. In order to illustrate the operation of the decimation filters, assume that we need to decimate a video signal from a sensor with 1000×1000 pixels operated at sample rate of 2kHz. Also assume an oversampling ratio of 64, an I/O bus width of 64 bits, and 8 decimation filters. During each clock cycle, i.e. 500μs, the image sensor generates one bit plane. This data is transported from the pixels to the edge of the chip one row at a time. Each row is multiplexed between the decimation filters. When the decimation filters receive each set of 8 pixel bits they load their registers, from external memory, with the partial pixel sum from the last bit plane. Then the row data is multiplied[2] by the filter coefficient for that bit plane, and the result is stored in memory. The filter coefficients for the decimation filter change once every clock cycle, and there are 128 coefficients used cyclically. Since we must calculate two frames at one time the last operation, i.e. loading the register and performing a multiplication, is repeated once again. Finally, we must determine the the I/O bandwidth requirements, and the required decimation filter speed. The I/O bandwidth must be at least the number of bits per frame times the frequency of operation times the number frames that must be calculated in parallel divided by the bus width, i.e. $\frac{8 \times 1000 \times 1000 \times 2000 \times 2}{64} \approx 500$Mhz. The required speed of the decimation filters is the same as the I/O bandwidth 500Mhz.

[2]Note that one of the operands is only one bit, therefore the multiplication can be preformed using a single adder.

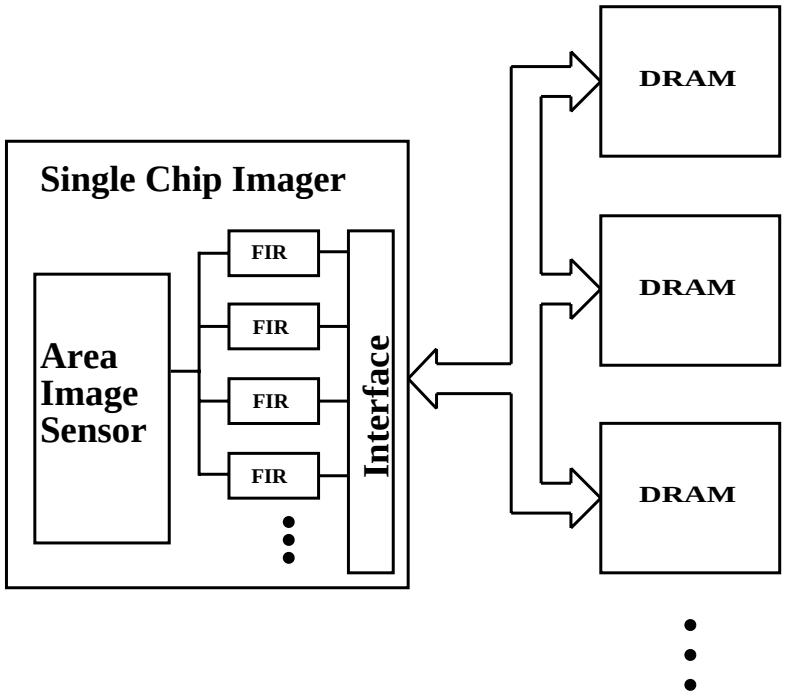


Figure 8.4: System block diagram of singe stage decimation filter integrated with an image sensor.

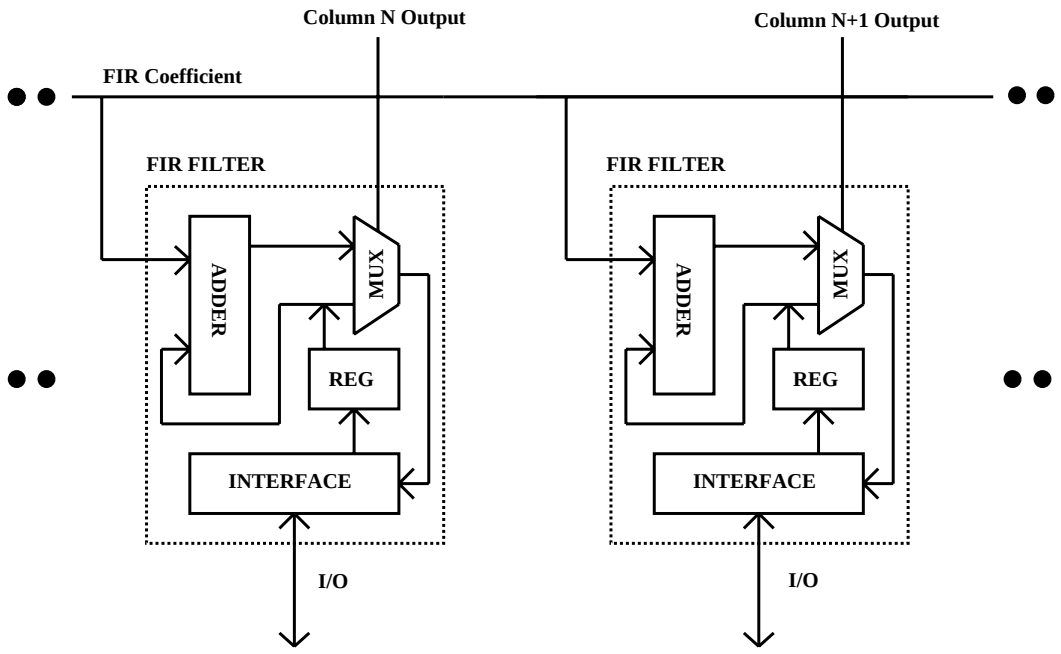


Figure 8.5: Block diagram of parallel decimation filter circuit.

8.4 Single Stage FIR Decimation Results

In order to test the linear estimation technique presented in the last section we simulated it and compared it with three other linear decimation filters. These included an FIR filter generated using the windowing method [52] with a Hamming window [52] and a 3dB bandwidth of 15Hz, a triangularly weighted FIR filter, and an FFT based filter. The FFT filter takes all of the data generated by the sigma delta modulator performs a fast Fourier transform [52] zeros all frequency components above 15Hz, and then performs the inverse fast Fourier transform. The FFT filter has the widest possible decimation window, the narrowest transition band, and the highest attenuation in the stop band of any linear filter. Each of the FIR filters simulated in this section use $2L$ taps.

Two separate video clips, sampled at 30Hz, were used for the simulation. The video clips contain 150 frames, each frame contains 352×240 pixels, and each pixel is represented with 16 bits. At random we selected 256 pixels from each video clip for

our simulation. This represents $2 \times 256 \times 150 = 76800$ data points, or 512 streams of 150 samples. A histogram and the power spectral density of this data is shown in Figure 8.6. Using this data we calculated the average SNR as a function of the oversampling ratio. The results of this simulation are shown in Figure 8.7. Note that the FFT technique outperforms the linear estimation technique by 5-10dB. This is caused by the narrow effective decimation window of the linear estimation filter. Note that the SNR of the windowed FIR and triangularly weighted FIR flatten out at higher oversampling ratios. This is caused by passband distortion, because of the narrow decimation window.

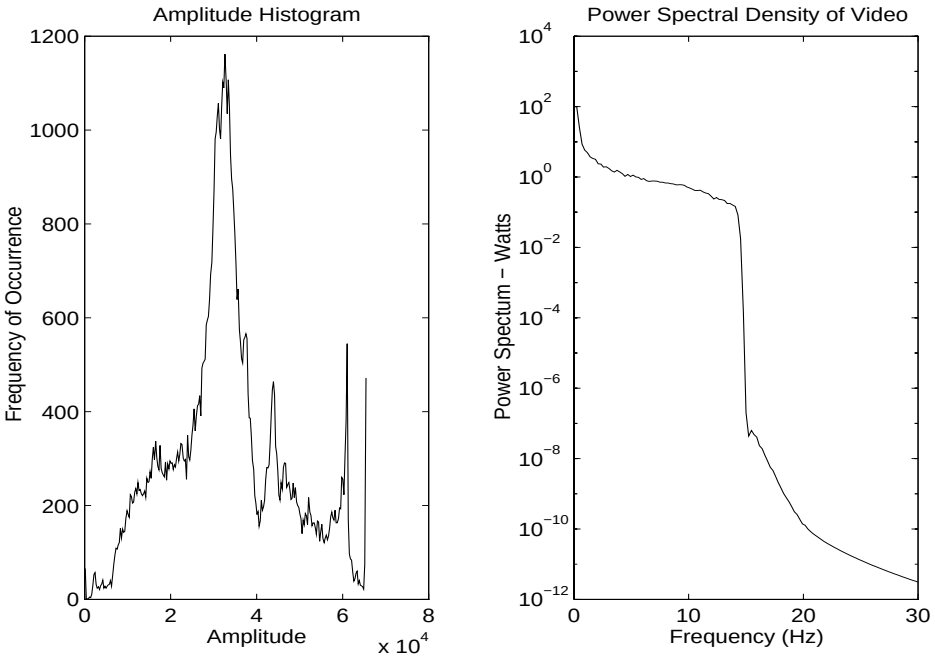


Figure 8.6: Histogram of data used for the simulation.

8.5 Summary and Discussion

We have discussed several nonlinear and linear decimation techniques for video data. We have found that single stage linear FIR decimation filters are best suited to our

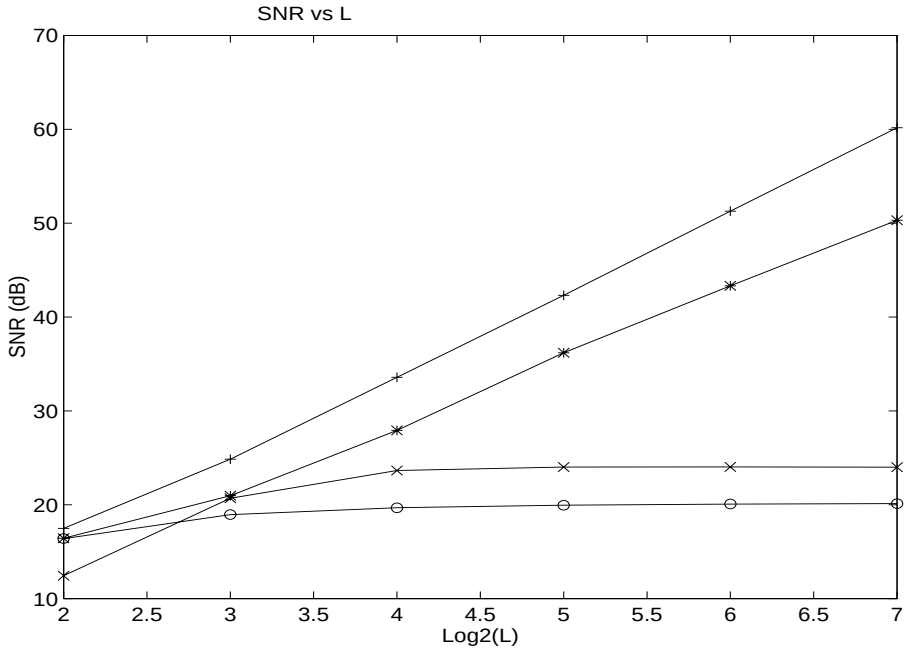


Figure 8.7: Simulation results

application. The goal of this work was to find a filter that could be implemented on the same chip with our image sensor and achieved an acceptable average SNR without an excessive amount of external memory. We also presented a single stage FIR filter design based on linear estimation theory and showed how well it performed on actual video data. This design also described how the decimation filter would be integrated with a video system. We also pointed out a few of the limitation of this system including high I/O bandwidths.

Chapter 9

Quadtree Based Lossless Image Compression

9.1 Introduction

In the last two chapters we discussed the digital signal processing necessary to decimate and filter the sigma delta modulated data generated by our image sensor. In this chapter we discuss another form of digital processing, lossless bilevel image compression. We are interested in reducing the amount of information that must be transmitted from the image sensor without decimation. In order to achieve this goal we investigated bilevel, or bit plane based, compression algorithms that could be implemented on the same chip with an image sensor.

The first compression algorithm we investigated for integration with our image sensor is based on the well known quadtree data structure [67]. This data structure is the two dimensional extension of a balanced binary tree. Each node of the tree has four descendents, and the leaves of the tree are the image pixels. As exemplified in Figure 9.1, the quadtree algorithm we employeed involves performing logical OR operations on blocks of pixel data to create pixels in successive layers of a resolution pyramid. This is a form of fixed to variable length coding that exploits background skipping. Such simple logical operations can be easily performed during the "read out" from a sensor without any impact on access time. As a result, this algorithm

is well suited for integration with a sensor. It exploits the parallelism available on a chip without requiring complex circuitry. Unfortunately, the quadtree algorithm does not provide adequate compression ratios, providing typically less than a factor of 2, compared to ratios of 20 to 100 for state of the art algorithms such as the JBIG standard [39]. JBIG and the rest of the acronyms used in this chapter are defined in Table 9.1. To achieve maximal compression we considered the *basic*, *non-progressive* JBIG. This algorithm does not, however, take full advantage of the parallelism afforded by integration since only local image information is used during the coding process.

Abbreviation	Definition
JBIG	Joint Bilevel Image experts Group
ABIC	Adaptive Bilevel Image Compression
RR	Resolution Reduction
DP	Deterministic Prediction
TPB	Typical Prediction - Base Layer
TPD	Typical Prediction - Differential Layer
PRES	A suggested JBIG resolution reduction method, with matched deterministic prediction method implied.
QT	Quadtree Algorithm - A compression algorithm interpretable as a JBIG matched resolution reduction method and deterministic prediction method.

Table 9.1: Definitions of abbreviated compression terms.

To get the best of both worlds we decided to use adaptive arithmetic coding with conditional prediction, as used in JBIG, following quadtree compression. This idea with a few variations generated three different quadtree algorithms. We then realized that one of these algorithms was compliant with the JBIG standard! The quadtree data structure can be viewed as a *resolution reduction* pyramid in a *progressive* JBIG version. The quadtree algorithm behaves like a combination of a resolution reduction method and the *deterministic prediction* [61] option in a progressive JBIG algorithm.

Deterministic prediction skips over pixels in coding that can be completely predicted at a decoder, given only the already reconstructed pixels of the progressive resolution pyramid. Such a quadtree (QT) algorithm can be completely realized within

the constraints of the JBIG user-specifiable resolution reduction method, along with its corresponding constrained table for deterministic prediction. Note that the deterministic prediction implied by the quadtree algorithm does not completely exploit all of the deterministic prediction potential presented by its resolution reduction scheme. This will be explained further in Subsection 9.2.1.

We began this research by trying to compress bit planes from our image sensor, but we quickly realized that this was difficult without decimating the data first. This is shown in Section 9.3. Although, compression of sigma delta modulated bit planes is difficult, the algorithms we developed efficiently compressed standard bilevel images such as FAX documents.

We begin this chapter by presenting our contributions to bilevel image compression research. Then we conclude by presenting the difficulties we encountered while trying to compress sigma delta modulated bit planes. In Section 9.2 we describe our JBIG compliant QT algorithm, and compared it with other JBIG variations on a set of 8 CCITT FAX images. In Section 9.3 we describe the other two QT algorithms, and compared all three algorithms with non-progressive JBIG on both grey coded and sigma delta modulated images from the USC data base.

9.2 JBIG Bilevel Image Compression

A quadtree based JBIG compliant bilevel image compression algorithm is described. In terms of the number of arithmetic coding operations required to code an image, this algorithm is significantly faster than previous JBIG algorithm variations. Based on this criterion, our algorithm achieves an average speed increase of more than 9 times with only a 5% decrease in compression when tested on the eight CCITT bilevel test images and compared against the basic non-progressive JBIG algorithm. The fastest JBIG variation that we know of, using "PRES" resolution reduction and progressive buildup, achieved an average speed increase of less than 6 times with a 7% decrease in compression, under the same conditions.

This section is organized as follows. The next subsection provides a brief review of the basic JBIG algorithm, hierarchical resolution reduction of binary images, and

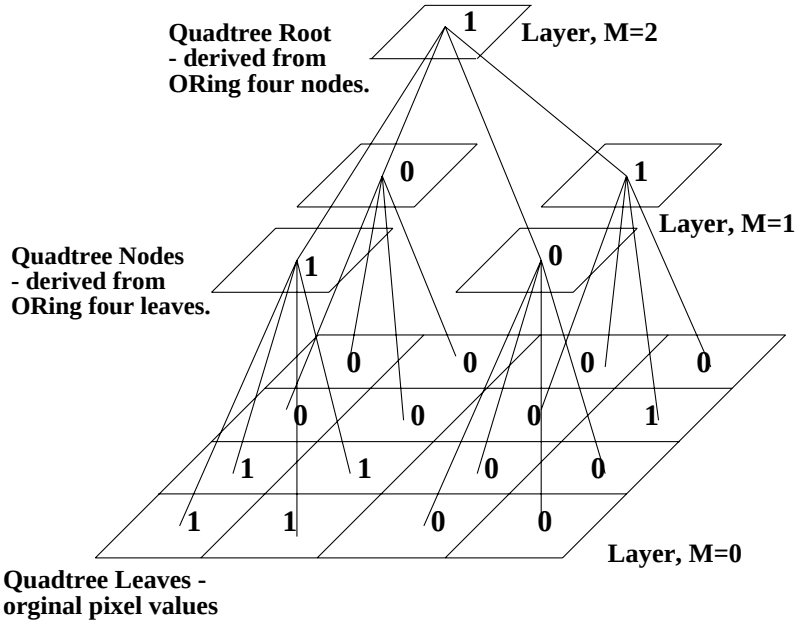


Figure 9.1: Quadtree Algorithm.

typical and deterministic pixel prediction. Our JBIG compliant quadtree (QT) algorithm is described in Subsection 9.2.2. In Subsection 9.2.3 we use a set of eight CCITT images to compare our algorithm to other versions of JBIG algorithms on the basis of compression and speed. We show that our algorithm is uniformly faster in terms of arithmetic coding operations and achieves comparable compression.

9.2.1 Algorithmic Aspects of JBIG

JBIG compression algorithms can be categorized into two classes: non-progressive and progressive, where in the latter case a hierarchy of resolution information about the image is available for coding. Information on the basic non-progressive JBIG to which we also compare our QT JBIG algorithm can be found in [39].

A block diagram of a progressive JBIG binary image compression encoder is given in Figure 9.2. The algorithm begins by transforming the original image into a pyramid of hierarchical, resolution reduced images. Each resolution reduced image is then encoded, beginning with the lowest resolution image. Typical and deterministic

prediction [61] of pixels, as will be explained, is used where possible, so that such pixels can be skipped in the arithmetic coding step – thus achieving better compression along with the desired compression speed up. The unskipped pixels are arithmetically coded as described in [39].

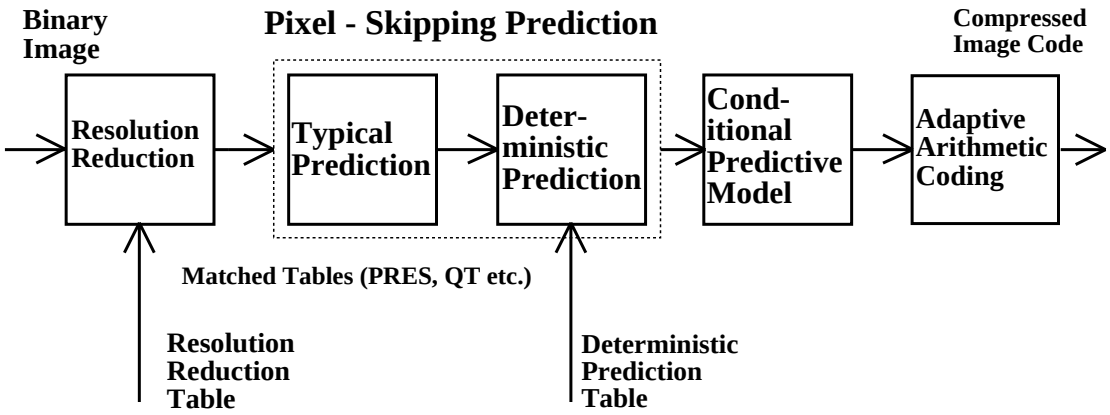


Figure 9.2: Block Diagram of JBIG Compression Encoder

Hierarchical resolution reduction (RR) of binary images transforms a single image into a pyramid of progressively smaller images. In keeping with the JBIG standard, we assume that resolutions are reduced symmetrically by factors of two. For example, a 4×4 pixel image is transformed into a pyramid of three resolution reduced layers as shown in Figure 9.1. These three layers consist of the 4×4 pixel image (M = 0), a reduced 2×2 pixel image (M = 1), and a further reduced 1 pixel image (M = 2) respectively. Each *layer* of the resolution reduction pyramid is created using information from the previous higher resolution layer and possibly the present layer. Each *pixel* in the next lower resolution layer replaces four pixels in the higher resolution layer. Typically, the same method is used to create each successive layer of the resolution reduction pyramid.

The PRES resolution reduction method suggested in JBIG is more complicated than the QT method. Its purpose is to generate high quality, i.e. visually appealing,

lower resolution images. To determine the value of a resolution reduced pixel, the PRES method uses three pixels from the same layer and nine pixels from the previous higher resolution layer. A complete description of this method is given in [39]. In contrast to this, the quadtree reduction method uses only four higher resolution pixels.

Figure 9.3 illustrates results from the PRES and QT methods on a 256×256 pixel image, assuming five resolution reductions for both. The resolution reduced images in this figure, as well as in Figure 9.5, and 9.8 are shown with pixels representative of their resolution. Note that with QT reduction, successive lower resolution images rapidly become more black resulting in a faster loss of quality when compared with PRES reduction. Total illegibility of the QT RR images occurs about one resolution reduction sooner than that of the PRES RR images. Coding time, measured in terms of the number of arithmetically encoded pixels, is our design objective rather than the quality of resolution reduced images as was done previously.

Pixel prediction exploits the fact that hierarchical resolution reduction layers, being multiple representations of the same image, contain highly related and sometimes redundant information. Such prediction can be used to skip over and thus reduce the number of pixels to be arithmetically coded, thereby increasing compression speed.

Typical Prediction (TPB or TPD)

There are two methods for typical prediction: one tailored for progressive (TPD) and the other for non-progressive (TPB) JBIG [39]. To illustrate how these two methods differ, the PRES RR images of Figure 9.3 are encoded, and the arithmetically encoded pixels are displayed in black in Figure 9.4. For TPD, the progressive pyramid of black, arithmetically coded pixels for all resolution layers is illustrated. Here, we have also included the arithmetically encoded pixel skipping effects of deterministic prediction and labeled the results accordingly TPD/DP. Although there are 6 pyramid layers, it is evident that their black pixels are collectively fewer than those within the non-progressive D0-TPB or D0 pseudo images.

To determine if a 2×2 *quad* of higher resolution pixels is typical or not, TPD [39] uses a 3×3 neighborhood of lower resolution pixels. A quad is defined to be typical if all the pixels in it and all the pixels in the nine nearest lower resolution neighbors

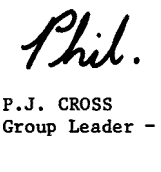
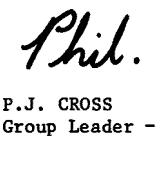
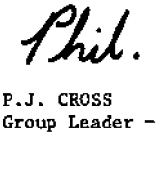
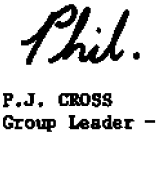
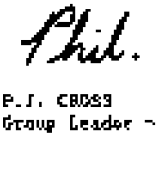
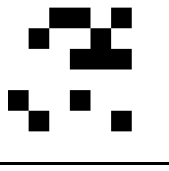
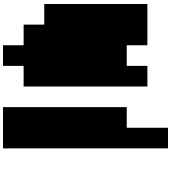
Resolution Reduction using the PRES or QT Methods				
dpi	PRES	% Black	QT	% Black
200		11.18		11.18
100		11.06		13.85
50		11.67		18.43
25		13.09		25.49
12.5		16.02		41.02
6.25		18.75		57.81

Figure 9.3: Resolution Reduced Images using the PRES or QT Methods.

have the same value. For example, a quad is non-typical if the nine nearest neighbors have the same value but one or more of its four higher resolution pixels differ from that value. A higher resolution line-pair of pixels is non-typical, if it contains *any* non-typical quads. If a *line-pair* of pixels is typical, then all of its typical quads can be skipped during arithmetic encoding.

When TPD is specified during encoding and decoding, a flag bit is used to indicate typical behavior for each line pair in each image of the resolution reduction pyramid, except that the lowest resolution layer is encoded using TPB. Although these flag bits typically have a very skewed distribution, i.e. they are easy to compress, they do reduce the compression ratio. As such, TPD trades the coding of a number of predictable pixels for the coding of a few flag bits. The second column in Figure 9.5 illustrates the pixels skipped using TPD for the PRES RR images of Figure 9.3. In contrast to Figure 9.3, the black pixels in Figure 9.5 are pseudo-images that represent the arithmetically encoded pixels.

TPB uses the previous scan line to predict all the pixels in the next scan line of image data. A flag bit is encoded at the beginning of each line to indicate whether "typical" prediction held true for that line. This figure, as well as Figure 9.8, again shows the digitally reduced images magnified to original image size to facilitate comparisons.

Deterministic Prediction (DP).

DP [39] uses information from lower resolution pixels to *exactly* determine the value of the next higher resolution pixels. Since the DP rules are based on inversion of the RR method, these methods must be matched. Any higher resolution pixels that can be exactly determined are again not arithmetically encoded. For example, assume that the RR method used determines each lower resolution pixel by finding the logical "AND" of the four associated higher resolution pixels. Then, if a lower resolution pixel has a value of one, its corresponding higher resolution pixels also have values of one and therefore can be predicted. The last column in Figure 9.5 illustrates the arithmetically encoded pixels using DP for the PRES RR images. Notice how few of

the arithmetically encoded pixels are skipped with DP, that those skipped are concentrated near the edges and that they complement the pixels skipped using TPD. Although few in number, DP skipped pixels contribute more heavily to compression gains. The edge pixels skipped by DP, contain more information than the background pixels skipped using TPD.

Typical and Deterministic Prediction (TPD/DP)

When TPD and DP are combined, they are commutative and can essentially be performed in parallel, i.e. any pixel that is typically predicted is not arithmetically encoded and any pixel that is deterministically predicted is also not arithmetically coded. The fourth column of Figure 9.5 illustrates the arithmetically encoded pixels using TPD/DP for the PRES RR images.

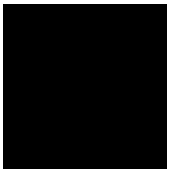
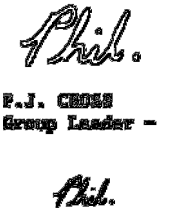
Arithmetically Coded Pixels for TPB or PRES TPD/DP Prediction						
dpi	D0	% Coded	D0 TPB	% Coded	D5 PRES TPD/DP	% Coded
200		100.00		56.25		15.57
100						24.44
50						36.43
25						50.20
12.5						69.92
6.25						100

Figure 9.4: Arithmetically Coded Pixels Using Various JBIG Methods

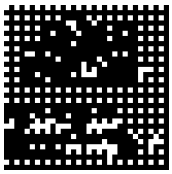
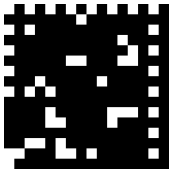
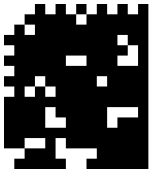
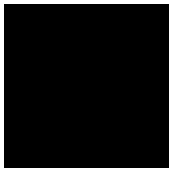
Arithmetically Coded Pixels for PRES Reduction - TPD, DP or TPD/DP Prediction						
dpi	D5 PRES TPD	% Coded	D5 PRES DP	% Coded	D5 PRES TPD/DP	% Coded
200		18.16		77.64		15.57
100		28.52		78.13		24.44
50		42.48		79.52		36.43
25		61.72		78.71		50.20
12.5		84.38		81.64		69.92
6.25		100		100		100

Figure 9.5: Arithmetically Coded Pixels for PRES Reduction using TPD, DP or TPD/DP Prediction

9.2.2 JBIG Compliant Quadtree Algorithm

The QT algorithm can be implemented using the JBIG standard with a user defined RR table, and a matched DP table. Therefore to completely specify QT it is sufficient to describe the RR and the DP methods.

The QT RR method uses a logical OR of four higher resolution pixels to determine each lower resolution pixel, i.e. if all four higher resolution pixels are zero then the corresponding lower resolution pixel is zero, otherwise it is a one. Pseudo code for the method is shown in Figure 9.6.

```
QuadtreeResolutionReduction(BinaryQuadtree) {
   i = 0;
   While (i < NUMBER OF RR LAYERS) {
      j = 0;
      While (j < (Y DIMENSION OF ORIGINAL IMAGE) / 2^(i+1)) {
         k = 0;
         While (k < (X DIMENSION OF ORIGINAL IMAGE) / 2^(i+1)) {
            BinaryQuadtree(i+1,j,k) = BinaryQuadtree(i,j*2,k*2) OR
            BinarQuadtree(i,j*2+1,k*2) OR
            BinaryQuadtree(i,j*2,k*2+1) OR BinaryQuadtree(i,j*2+1,k*2+1);
            k = k + 1;
         }
         j = j + 1;
      }
      i = i + 1;
   }
}
```

Figure 9.6: Quadtree Resolution Reduction Method

The arithmetic coder following the QT DP method skips the coding of higher resolution pixels if their corresponding lower resolution pixel is zero. Therefore, any time a pixel with a value of zero is encountered during arithmetic encoding all corresponding higher resolution pixels need not be coded since all their values are zero. This is true by construction because of the QT RR method. Pseudo code for the

algorithm is shown in Figure 9.7.

```
QuadtreeDeterministicPrediction(BinaryQuadtree) {
   i = NUMBER OF RR LAYERS-1;
   While (i >= 0) {
      j = 0;
      While (j < (Y DIMENSION OF ORIGINAL IMAGE) / 2^(i+1)) {
         k = 0;
         While (k < (X DIMENSION OF ORIGINAL IMAGE) / 2^(i+1)) {
            if (i NOT = NUMBER OF RR LAYERS-1) {
               if (BinaryQuadtree(i+1,j/2,k/2) = 0) {
                  SKIP PIXEL;
               }
               else {
                  ARITHMETICALLY ENCODE PIXEL;
               }
            }
            else {
               ARITHMETICALLY ENCODE PIXEL;
            }
            k = k + 1;
         }
         j = j + 1;
      }
      i = i - 1;
   }
}
```

Figure 9.7: Quadtree Deterministic Prediction Method

The QT DP method does not fully exploit the information in the RR pyramid. For example, if three higher resolution pixels in a given quad are zero but the associated lower resolution pixel is one then the remaining pixel is known to be one. This addition to the RR method would reduce the total number of pixels that are arithmetically encoded, but would slow down a hardware realization because of the required additional pixel "reads". Since this type of prediction increases the speed

by less than 1% and decreases the compression by 0.5%, the cost of additional pixel reads outweighs the speed benefits obtained.

Figure 9.8 illustrates the operation of TPD and DP with QT RR using the QT images from Figure 9.3. Note the horizontal stripes in the D5 QT TPD column, which are caused by non-typical line pairs in the hierarchical image. One can visualize the superior pixel-skipping achieved using QT, by comparing the D5 PRES TPD/DP column of Figure 9.5 with the rightmost, D5 QT TPD/DP, column of Figure 9.8.

Figure 9.9 visually illustrates the difference between the pixel prediction capabilities of PRES and QT.

9.2.3 Relative Speed and Compression of QT

The algorithms' speed and compression are compared. We define algorithm speed to be inversely proportional to the number of pixels presented to the arithmetic coder. To benchmark our fastest JBIG compliant QT compression algorithm (D5 QT-TPD/DP) we compare it to the basic non-progressive JBIG algorithm D0 known to achieve the best compression, and to the suggested progressive JBIG algorithm using PRES (D5 PRES-TPD/DP) which previously achieved the best speed up known from skipping arithmetically coded pixels. This comparison is done using a set of eight CCITT test images. Each image has 1728×2376 pixels, but because of software limitations the y dimension was reduced to 2304 by removing the last 72 lines. This results in 1728×2304 images containing 3,981,312 pixels. The removed lines in each image compress greatly since they are of uniform color and thus have relatively little impact on the results.

Figure 9.10 compares the relative speed of compression achieved by the D5 QT-TPD/DP, D5 PRES-TPD/DP, D0-TPB and D0 using D0 as a reference. Note that our QT algorithm is in all cases faster than the other three algorithms. On average, it is 1.6 times faster that the PRES algorithm, 6.8 times faster than the D0-TPB algorithm, and 9.4 times faster than the D0 algorithm used as a reference. The compression ratios of the four algorithms are compared in Figure 9.11. The D0-TPB algorithm achieves an average compression similar to that of the reference D0

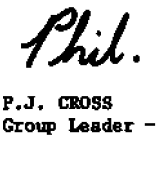
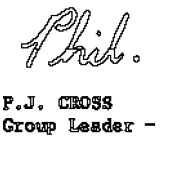
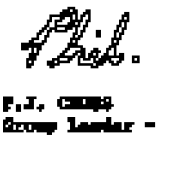
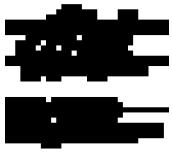
Arithmetically Coded Pixels for QT Reduction - TPD, DP, or TPD/DP Prediction						
dpi	D5 QT TPD	% Coded	D5 QT DP	% Coded	D5 QT TPD/DP	% Coded
200		19.74		77.64		9.74
100		36.89		78.13		16.43
50		56.45		79.52		25.10
25		77.73		78.71		39.83
12.5		100		81.64		57.81
6.25		100		100		100

Figure 9.8: Arithmetically Coded Pixels for QT Reduction using DP, TPD or TPD/DP Prediction

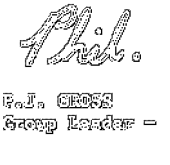
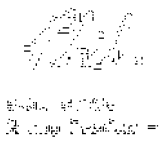
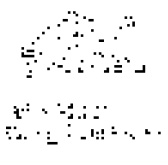
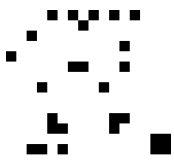
Arithmetically Encoded Pixels using DP/TPD				
dpi	PRES Only	% Black	QT Only	% Black
200		7.46		1.63
100		10.55		2.54
50		14.94		3.61
25		18.36		8.01
12.5		22.66		10.55
6.25		0.00		0.00

Figure 9.9: The number of nonpredicted pixels using PRES and QT is compared. The highest resolution layer is at the top of each column. Each layer has 4 times as many pixels as the layer below it. All of the images have also been scaled using to the same size using pixel replication. "PRES only" shows black pseudo pixels for pixels that must be encoded with PRES that must not be encoded with the QT method. "QT only" shows corresponding pseudo-images for the pixels that only need to be encoded with the QT method.

algorithm. Our algorithm achieves an average of 2% better compression than the PRES algorithm and 5% less compression than the D0 algorithm. Note that our algorithm also consistently achieves better compression than the PRES algorithm (D5 PRES-TPD/DP). Tables 9.2 and 9.3 contain the results used to generate the figures.

9.2.4 Conclusion

We described a JBIG compliant quadtree based compression algorithm, intended for integration with a bilevel area image sensor on the same chip. We demonstrated that our algorithm is significantly faster than basic non-progressive JBIG, and progressive JBIG using PRES, while achieving comparable compression. We believe that our algorithm may also lend itself to faster operation even when implemented in software.

Another form of parallelism utilizes multiple adders to increase the speed of arithmetic coding [57]. A hardware architecture displaying this type of parallelism was recently presented by Feygin, *et al.* [29]. Note that multiple adders occupy more silicon real estate than simple quadtree logic even when the quadtree logic must be replicated for every scan line of an area image sensor. Furthermore, this parallel arithmetic coding technique exploits the same contiguous image background regions as a quadtree front-end. We suspect that parallel arithmetic coding techniques will fail to increase coding speed if background pixels are already skipped over by a quadtree front-end. Since the two methods appear to be mutually exclusive, we chose to concentrate on the quadtree parallelism.

Comparison of our results with those of Feygin, *et al.* [29] is not straightforward, since they investigated speeding up ABIC [2] rather than JBIG and used a slightly different scan of the CCITT test images. Although ABIC is a direct precursor of the basic JBIG non-progressive algorithm, it has a smaller nearest neighbor model and is based on the "Q" [58] rather than the "QM" adaptive arithmetic coder [39]. Nevertheless, the results are quite consistent with our observation that both approaches exploit image background regions. Their reported average speed up, 7 to 13 times depending on complexity, is similar to our results. Moreover, the speed up per CCITT

test image is also quite similar.

We note that the CCITT test images used are representative of typical business documents, but do not include "digital halftones". We do not expect speed up performance increases for our algorithm or other algorithms mentioned here when applied to halftone images.

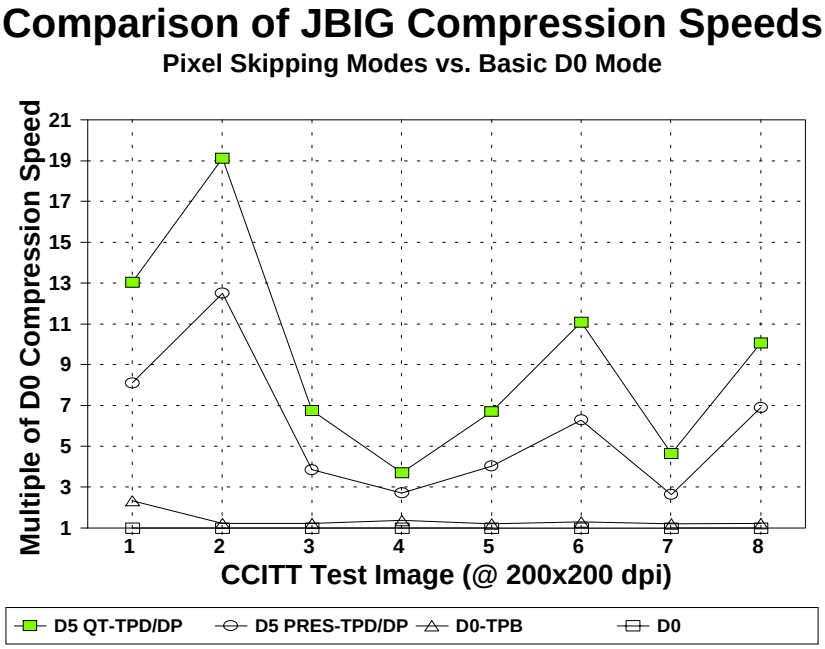


Figure 9.10: Comparison of JBIG Compression Speeds vs. Non-progressive JBIG

9.3 Sigma Delta Modulated Image Compression

Two QT based lossless image compression algorithms are described. These algorithms and the JBIG compliant QT algorithm are compared to non-progressive JBIG on a set of four gray scale and four sigma delta modulated images from the USC data base.

This Section is organized as follows. The Skipping and adaptive QT algorithms are described in Section 9.3.1. In Section 9.3.3 the three QT algorithms, skipping adaptive and JBIG compliant, are compared to the JBIG bilevel image compression standard [39] and experimental results are reported.

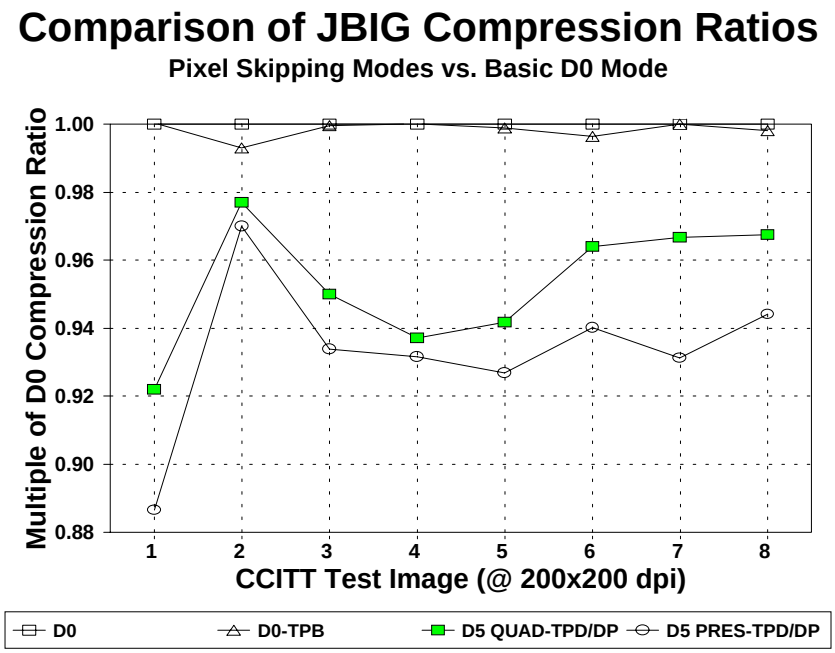


Figure 9.11: Comparison of JBIG Compression Ratios vs. Non-progressive JBIG

Arithmetically Coded Pixel Count For CCITT Images 1–8								
	D0	D0 TPB	D5 PRES TPD	D5 PRES DP	D5 PRES TPD/DP	D5 QT DP	D5 QT TPD	D5 QT TPD/DP
1	3981312	1703808	604472	4021440	490978	320616	1173044	305348
2	3981312	3252096	380464	4035178	318575	325348	851048	208152
3	3981312	3227904	1274300	4065718	1031557	637812	2171528	589192
4	3981312	2897856	1784552	4112969	1469132	1089676	2667868	1074352
5	3981312	3352320	1213000	4063504	987180	623752	2268912	594128
6	3981312	3077568	775636	4039549	633469	399960	1766076	359480
7	3981312	3326400	1876992	4087863	1514283	856240	3876300	854680
8	3981312	3305664	693096	4512112	576888	2312636	1642388	395876

Table 9.2: Results for Non-progressive JBIG, and QT or PRES Progressive JBIG

Compressed Image Size (Bytes) For CCITT Images 1–8								
	D0	D0 TPB	D5 PRES TPD	D5 PRES DP	D5 PRES TPD/DP	D5 QT DP	D5 QT TPD	D5 QT TPD/DP
1	14655	14650	17447	16526	16533	15787	16496	15895
2	8456	8515	9128	8734	8718	8563	9179	8655
3	21907	21916	24633	23432	23460	22893	23723	23061
4	53925	53921	60347	57949	57890	57320	58346	57546
5	25792	25823	29216	27805	27828	27194	28160	27389
6	12520	12566	13989	13294	13317	12850	13584	12987
7	56210	56211	64124	60280	60362	58062	59101	58146
8	14197	14223	15673	15038	15037	14625	15299	14674

Table 9.3: Results for Non-progressive JBIG, and QT or PRES Progressive JBIG

9.3.1 QT Compression Algorithms

Level Skipping

In an attempt to increase the speed and data compression of the JBIG Compliant QT algorithm, we explored skipping intermediate RR layers during coding. This algorithm is essentially the same as the JBIG compliant QT algorithm with the following change: after the highest resolution layer is coded the next lower resolution layer is skipped. All of the other lower resolution layers are coded in exactly same way as with the JBIG compliant QT algorithm. It is only the skipping of the second level of RR pyramid and the context used for highest resolution layer that distinguish this algorithms from the JBIG compliant QT algorithm.

The pixel context used for arithmetically coding the highest resolution layer is shown in Figure 9.12. The squares with x's are coded pixels in the highest resolution layer, and circles with x's are coded pixels in the third RR layer of the pyramid. The phase of each pixel in the highest resolution layer refers to the pixels location in each 4×4 block. Phase 1 refers to the four pixels in the upper left corner, phase 2 refers to the four pixels in the upper right corner, phase 3 refers to the four pixels in the lower left corner, and phase 4 refers to the four pixels in the lower right corner. The square with a question mark is the pixel to be coded. Several other contexts were tried, but they all produced lower compression ratios.

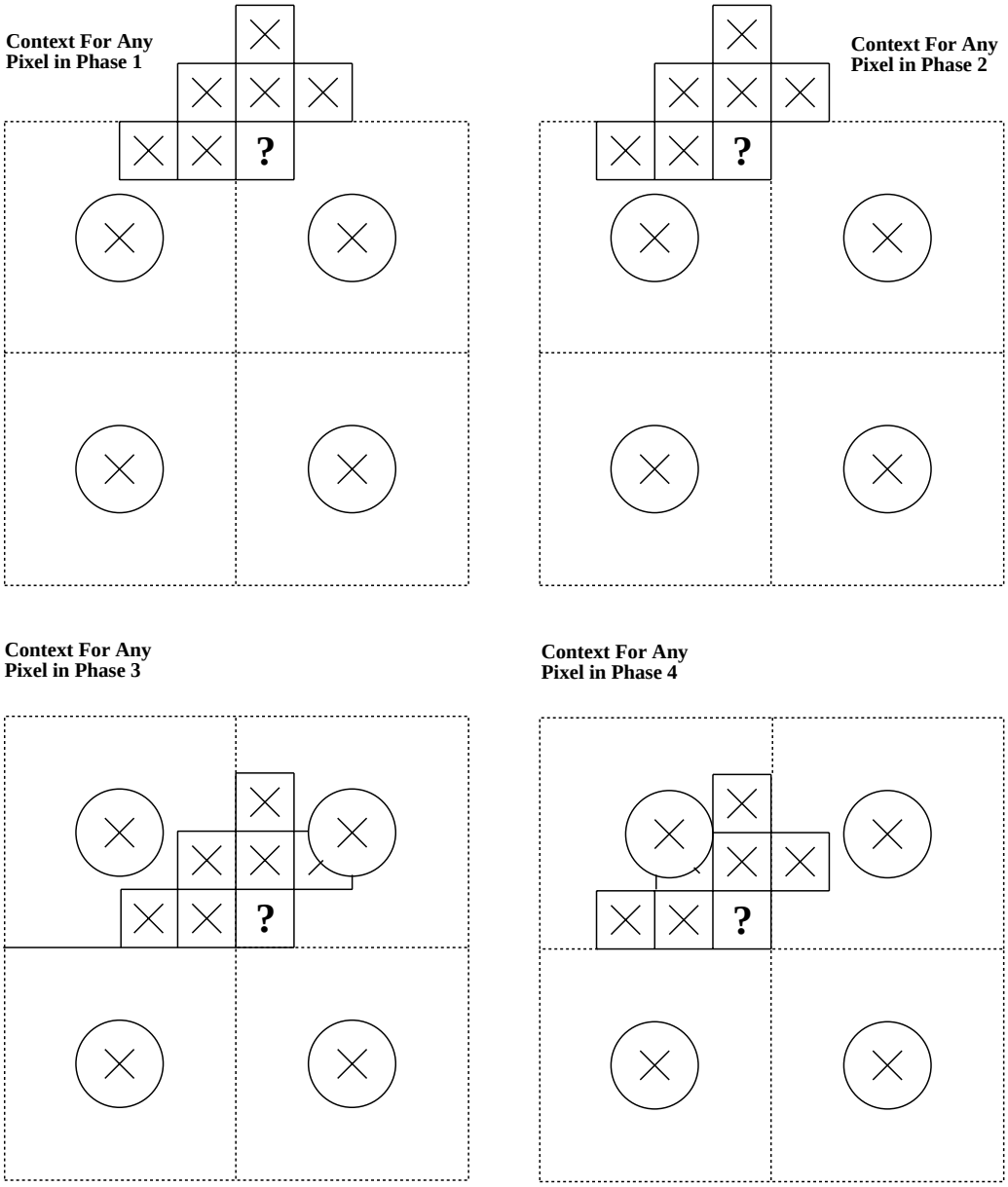


Figure 9.12: QT Skip Pixel Contexts For Level Zero

Since this algorithm skips certain quadtree levels it can not be directly implemented by the JBIG standard.

9.3.2 Adaptive

This algorithm is also very similar to the JBIG compliant QT algorithm except that the binary function used to create the RR images is adapted at each level of pyramid. There are sixteen different possible quad values, and therefore exactly sixteen different binary functions can be used for RR. The RR function used on each level is determined by finding the most probable quad type. When this type of quad is used to encode the next lower level resolution it is represented by a zero and all other quads are represented by a one. Quad skipping is performed identically to the methodology used in the JBIG compliant algorithm. Before coding each level of the tree the most probable quad must also be encoded so that the decoder will know what pixels can be skipped.

This algorithm can not be directly implemented by the JBIG standard because a new resolution reduction and deterministic prediction tables must be used for each level of the tree.

9.3.3 Results

The algorithms' speed and compression are compared to the non-progressive JBIG standard. We define algorithm speed to be inversely proportional to the number of pixels presented to the arithmetic coder. We define compression ratio to be the number of uncoded image bits divided by number of compression coded image bits. This comparison was done with four grey coded and four sigma delta modulated images from the USC data base. Each gray coded image consists of 8 512×512 bit planes, and each sigma delta modulated image consists of 32 512×512 bit planes.

The sigma delta modulated image data was generated using four 8 bit 512×512 gray scale images, and a first order 1 bit sigma delta modulator [12]. Figure 9.13 shows pseudo code for the algorithm used to convert each 8 bit gray scale image into 32 sigma delta modulated bitplanes. In Figure 9.13 GrayScaleImage is a two

```
MakeSigmaDeltaImageData(GrayScaleImage) {
   i = 0;
   While (i < 512) {
      j = 0;
      While (j < 512) {
         IntegratorValues[i][j] = 0;
         j = j + 1;
      }
      i = i + 1;
   }
   i = 0;
   While (i< 32) {
      Openfile(bitplane[i]);
      j = 0;
      While (j < 512) {
         k = 0;
         While (k < 512) {
            IntegratorValues[j][k] = IntegratorValues[j][k] +
            (GrayScaleImage[j][k]-128) - 128*Sgn(IntegratorValues[j][k]);
            BilevelImage[i][j][k] = (Sgn(IntegratorValues[j][k]) + 1)/2;
            If (i > 0) {
               Write (BilevelImage[i][j][k] XOR BilevelImage[i-1][j][k])
               to file bitplane[i];
            }
            Else {
               Write BilevelImage[i][j][k] to file bitplane[i];
            }
            k = k + 1;
         }
         j = j + 1;
      }
      Closefile(bitplane[i]);
      i = i + 1;
   }
}
```

Figure 9.13: Gray Scale Image to Sigma Delta Modulated Image Algorithm

dimensional array of bytes, IntegratorValues is a two dimensional array of floats, BilevelImage is a three dimensional array of bits, and Sgn is the signum function.

The results in Table 9.4 are based on compressing 32 bit planes, and the results in Table 9.5 are based on compressing 9 bit planes. 7 bits in Table 9.4 and 4 bits in Table 9.5 refer to the maximum achievable pixel SNR for the reconstructed sigma delta modulated data. In other words, if the sigma delta modulated pixel data was decimated using the optimal non-linear filter [36] the maximum number of bits necessary to represent each pixel would be ≤ 7 for the sigma delta modulated data used to generate Table 9.4 and ≤ 4 for the sigma delta modulated data used to generate Table 9.5.

32 Bit Plane (7 Bit SNR) Sigma Delta Modulated Data					
Algorithm	Image	Coded Bytes	Ratio	Coded Bits	Speed
JBIG	LAX	913481	1.15	8388608	1
JBIG QT	LAX	923987	1.13	10107904	0.83
QT Skip	LAX	913925	1.15	8686528	0.96
QT Adapt	LAX	922668	1.14	7796584	1.08
JBIG	Milkdrop	639687	1.64	8388608	1
JBIG QT	Milkdrop	645995	1.62	9259056	0.91
QT Skip	Milkdrop	636520	1.65	8084340	1.04
QT Adapt	Milkdrop	651220	1.61	6723524	1.25
JBIG	Sailing	877320	1.20	8388608	1
JBIG QT	Sailing	887523	1.18	9486552	0.88
QT Skip	Sailing	875895	1.20	8415036	0.99
QT Adapt	Sailing	891750	1.18	8004452	1.05
JBIG	Woman1	728482	1.44	8388608	1
JBIG QT	Woman1	738452	1.42	10388964	0.81
QT Skip	Woman1	730729	1.44	8774216	0.96
QT Adapt	Woman1	740471	1.42	6711356	1.25

Table 9.4:

The gray coded image data was generated using the same four 8 bit 512×512 gray scale images from the USC database using the algorithm shown in Figure 9.22.

Table 9.6 shows results for coding the 7 most significant binary images, and Table 9.7 shows results for coding the 4 most significant binary images, using similar precision for comparison with the above sigma delta results.

9 Bit Plane (4 Bit SNR) Sigma Delta Modulated Data					
Algorithm	Image	Coded Bytes	Ratio	Coded Bits	Speed
JBIG	LAX	197042	1.50	2359296	1
JBIG QT	LAX	200048	1.47	2447720	0.96
QT Skip	LAX	196942	1.50	2190160	1.08
QT Adapt	LAX	200540	1.47	1901284	1.24
JBIG	Milkdrop	91696	3.22	2359296	1
JBIG QT	Milkdrop	93781	3.14	2187536	1.08
QT Skip	Milkdrop	91617	3.22	1942932	1.21
QT Adapt	Milkdrop	94019	3.14	1516252	1.56
JBIG	Sailing	168446	1.75	2359296	1
JBIG QT	Sailing	171913	1.72	2268352	1.04
QT Skip	Sailing	168739	1.75	2066472	1.14
QT Adapt	Sailing	172450	1.71	1955496	1.20
JBIG	Woman1	131187	2.25	2359296	1
JBIG QT	Woman1	135175	2.18	2709244	0.87
QT Skip	Woman1	132597	2.22	2341868	1.01
QT Adapt	Woman1	134744	2.19	1536956	1.54

Table 9.5:

7 Bit PCM Data					
Algorithm	Image	Coded Bytes	Coded Ratio	Coded Bits	Speed
JBIG	LAX	165154	1.39	1835008	1
JBIG QT	LAX	167066	1.37	1914184	0.96
QT Skip	LAX	164799	1.39	1721136	1.07
QT Adapt	LAX	167620	1.37	1498208	1.22
JBIG	Milkdrop	99301	2.31	1835008	1
JBIG QT	Milkdrop	101234	2.27	1681388	1.09
QT Skip	Milkdrop	99663	2.30	1502004	1.22
QT Adapt	Milkdrop	101213	2.27	1296772	1.42
JBIG	Sailing	143861	1.59	1835008	1
JBIG QT	Sailing	146206	1.57	1952160	0.94
QT Skip	Sailing	144030	1.59	1725560	1.06
QT Adapt	Sailing	146732	1.56	1550680	1.18
JBIG	Woman1	131378	1.75	1835008	1
JBIG QT	Woman1	133980	1.71	2022916	0.91
QT Skip	Woman1	132190	1.74	1783628	1.03
QT Adapt	Woman1	134421	1.71	1420728	1.29

Table 9.6:

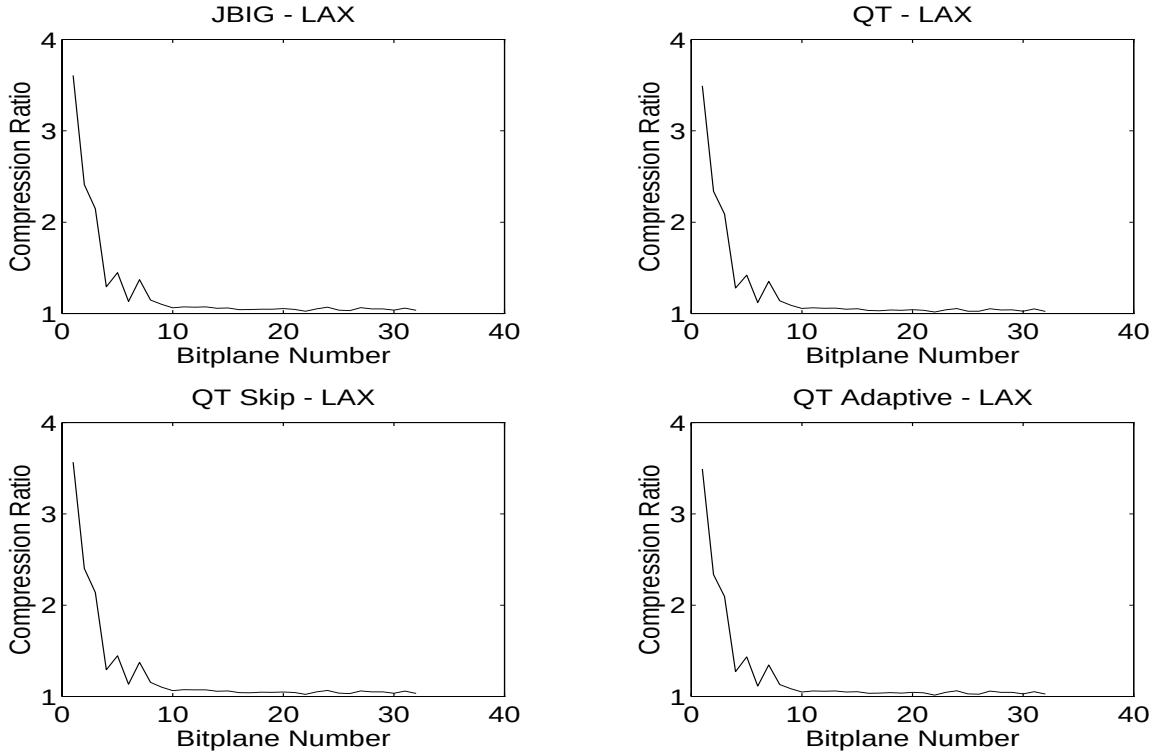


Figure 9.14: The image LAX is sigma delta modulated, the compression ratio of each algorithm is shown

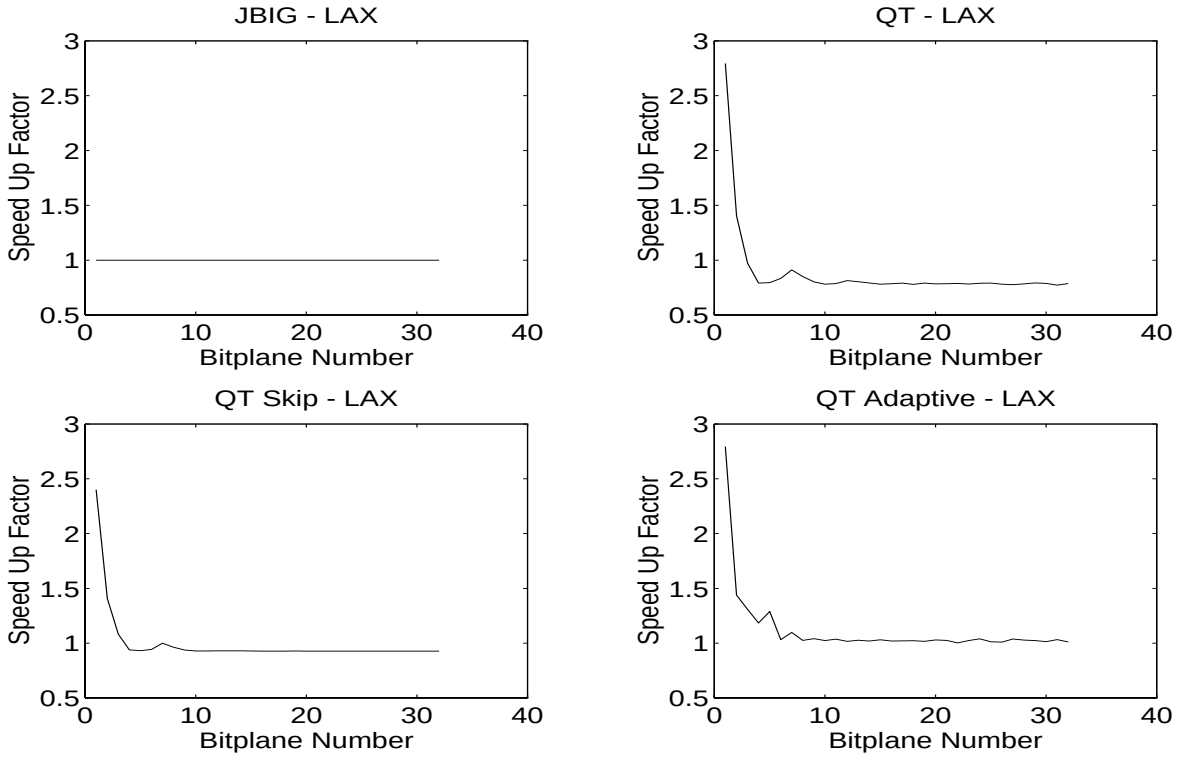


Figure 9.15: The image LAX is sigma delta modulated, the relative speed of each algorithm is shown

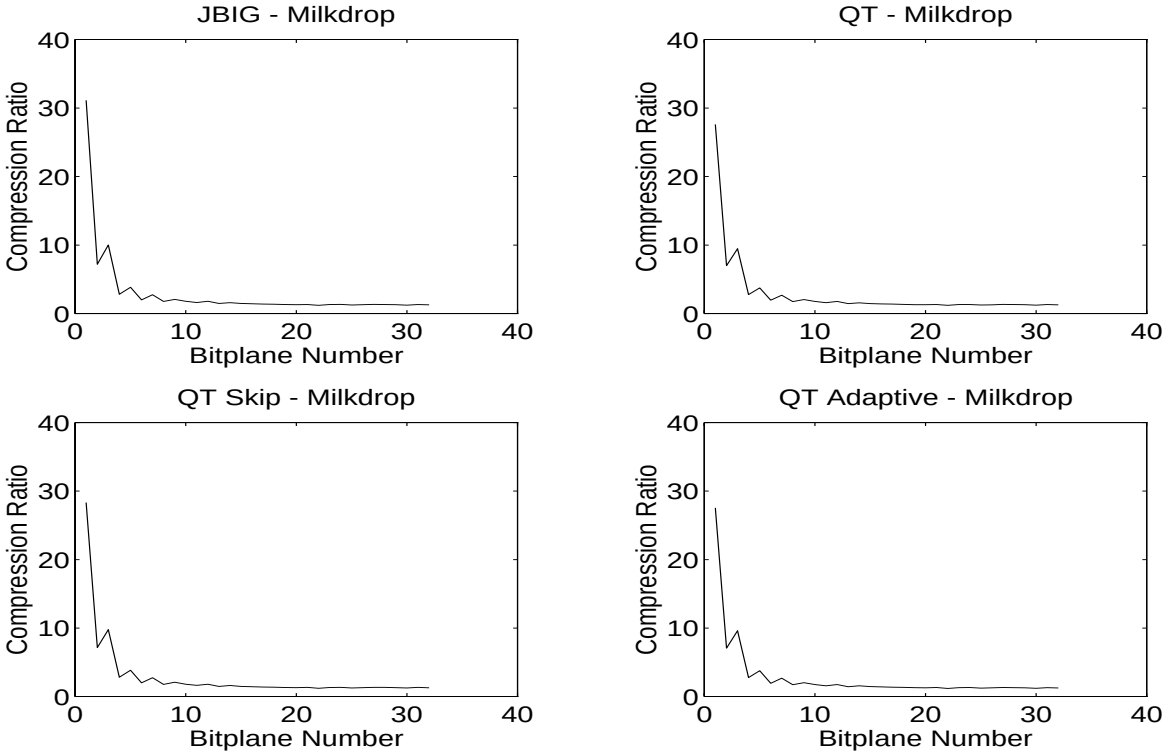


Figure 9.16: The image Milkdrop is sigma delta modulated, the compression ratio of each algorithm is shown

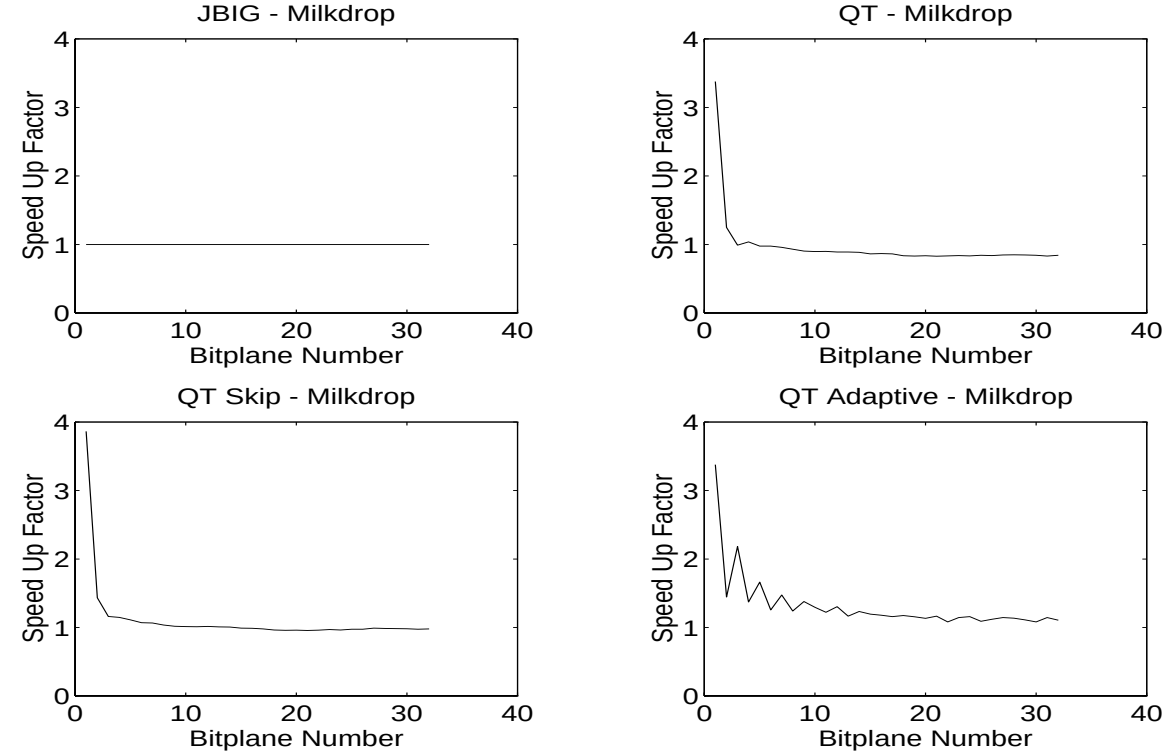


Figure 9.17: The image Milkdrop is sigma delta modulated, the relative speed of each algorithm is shown

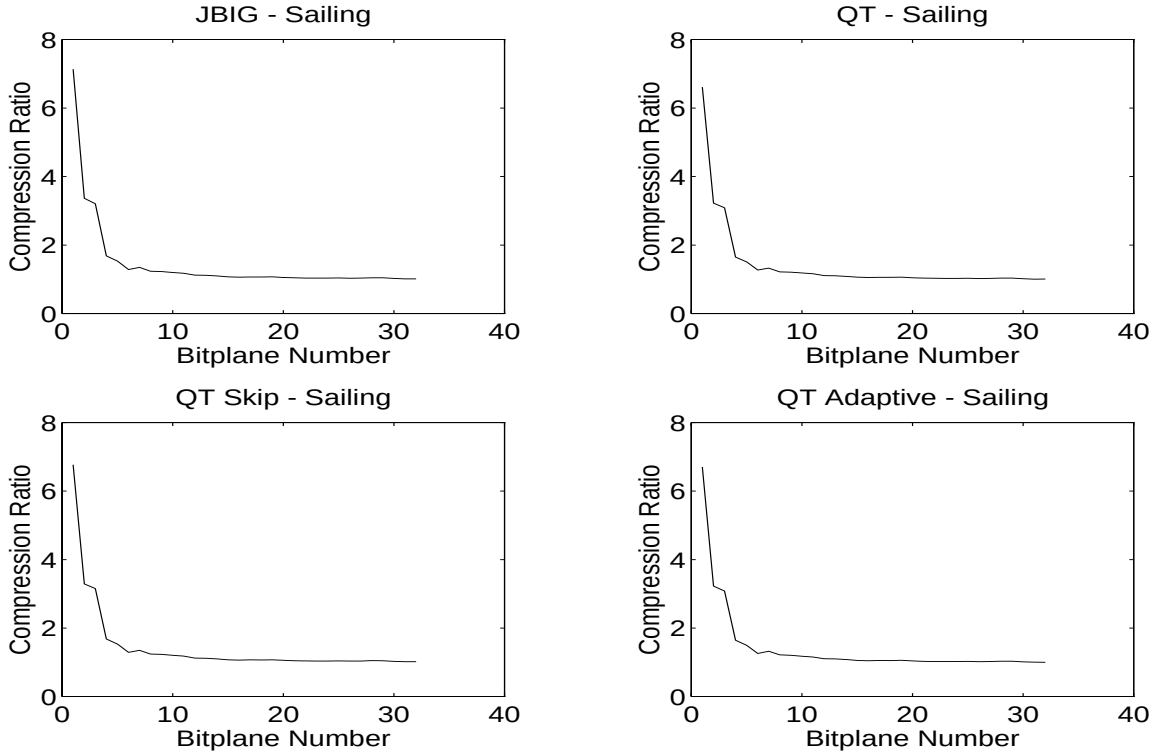


Figure 9.18: The image Sailing is sigma delta modulated, the compression ratio of each algorithm is shown

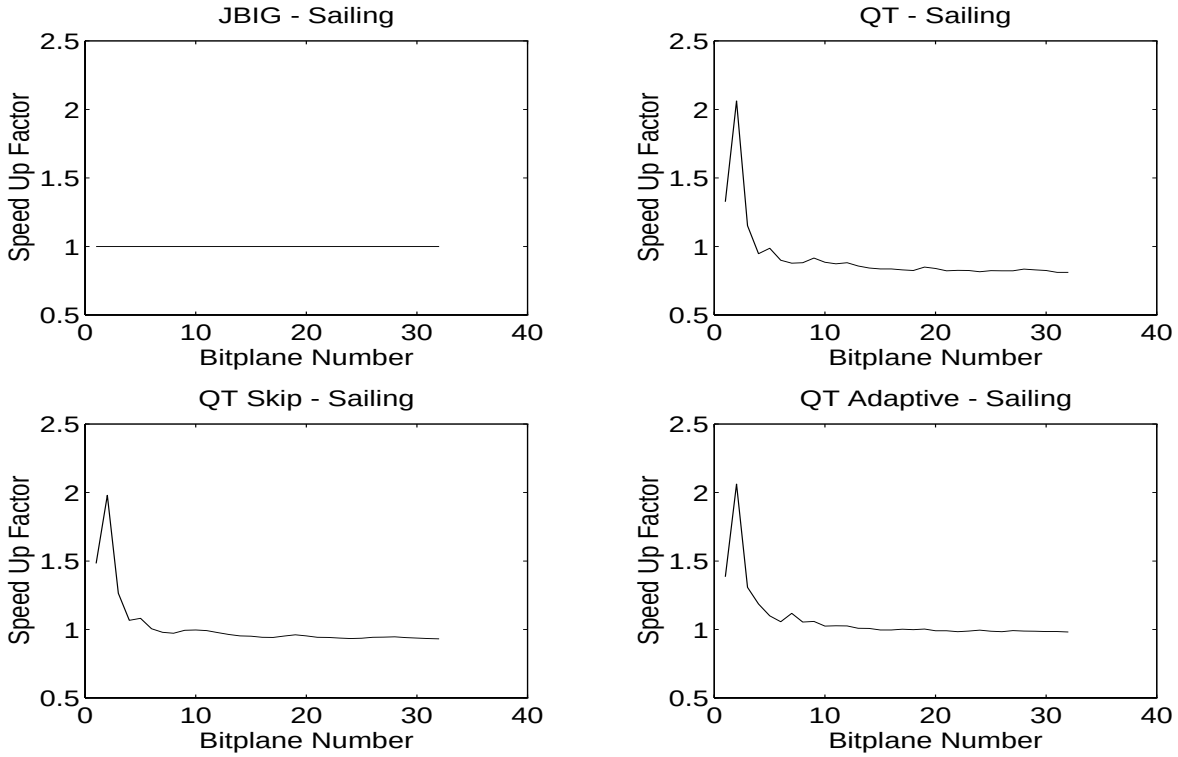


Figure 9.19: The image Sailing is sigma delta modulated, the relative speed of each algorithm is shown

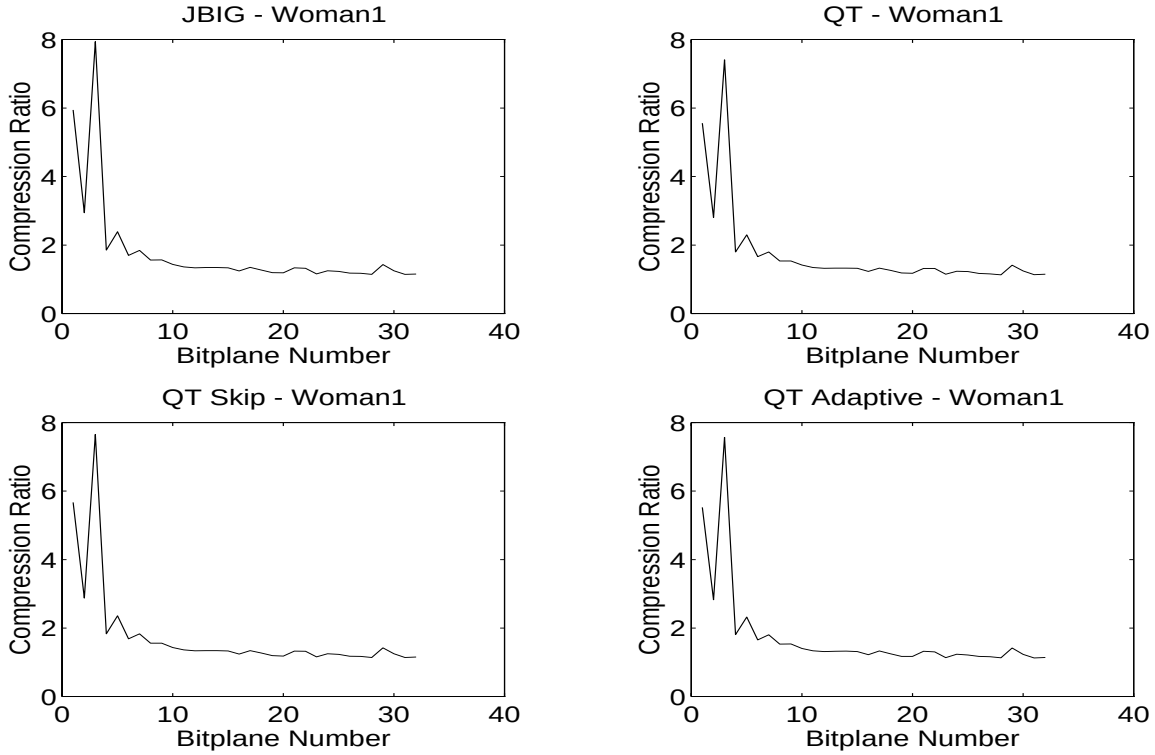


Figure 9.20: The image Woman1 is sigma delta modulated, the compression ratio of each algorithm is shown

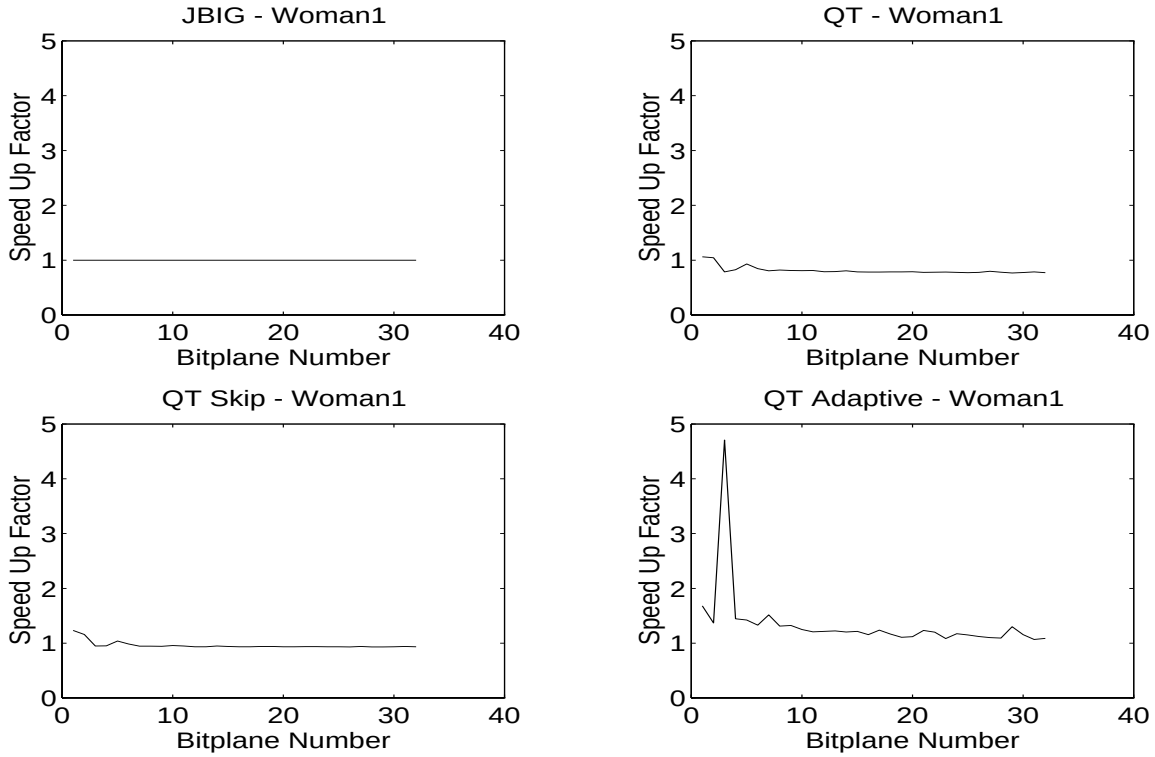


Figure 9.21: The image Woman1 is sigma delta modulated, the relative speed of each algorithm is shown

```
MakeGrayCodedImageData(GrayScaleImage) {
   i = 0;
   While ( i < 8 ) {
      Openfile(bitplane[i]);
      j = 0;
      While (j < 512) {
         k = 0;
         While (k < 512) {
            If (i NOT EQUAL 7) {
               Write (((GrayScaleImage[j][k] shift byte left i bits) XOR
               (GrayScaleImage[j][k] shift byte left i+1)) AND 1)
               to file bitplane[i];
            }
            Else {
               Write (GrayScaleImage[j][k] shift byte left i bits)
               to file bitplane[i];
            }
            k = k + 1;
         }
         j = j + 1;
      }
      Closefile(bitplane[i]);
      i = i + 1;
   }
}
```

Figure 9.22: Gray Scale Image to Gray Coded Binary Images Algorithm

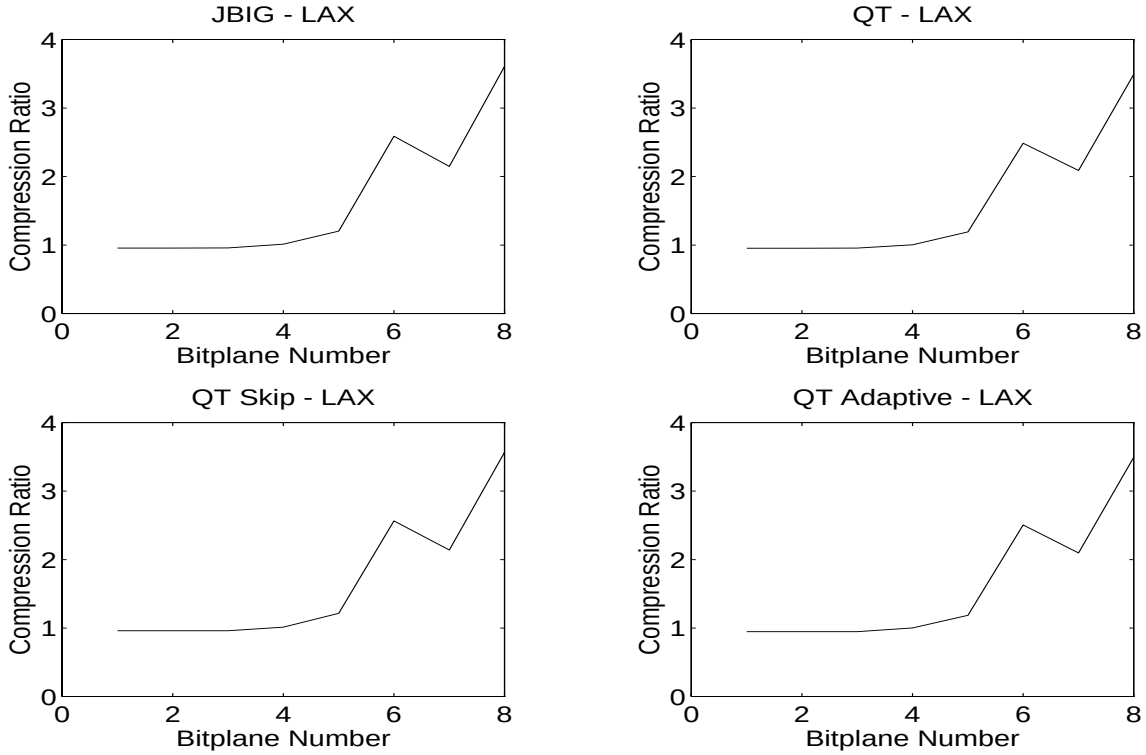


Figure 9.23: The image LAX is PCM gray coded, the compression ratio of each algorithm is shown

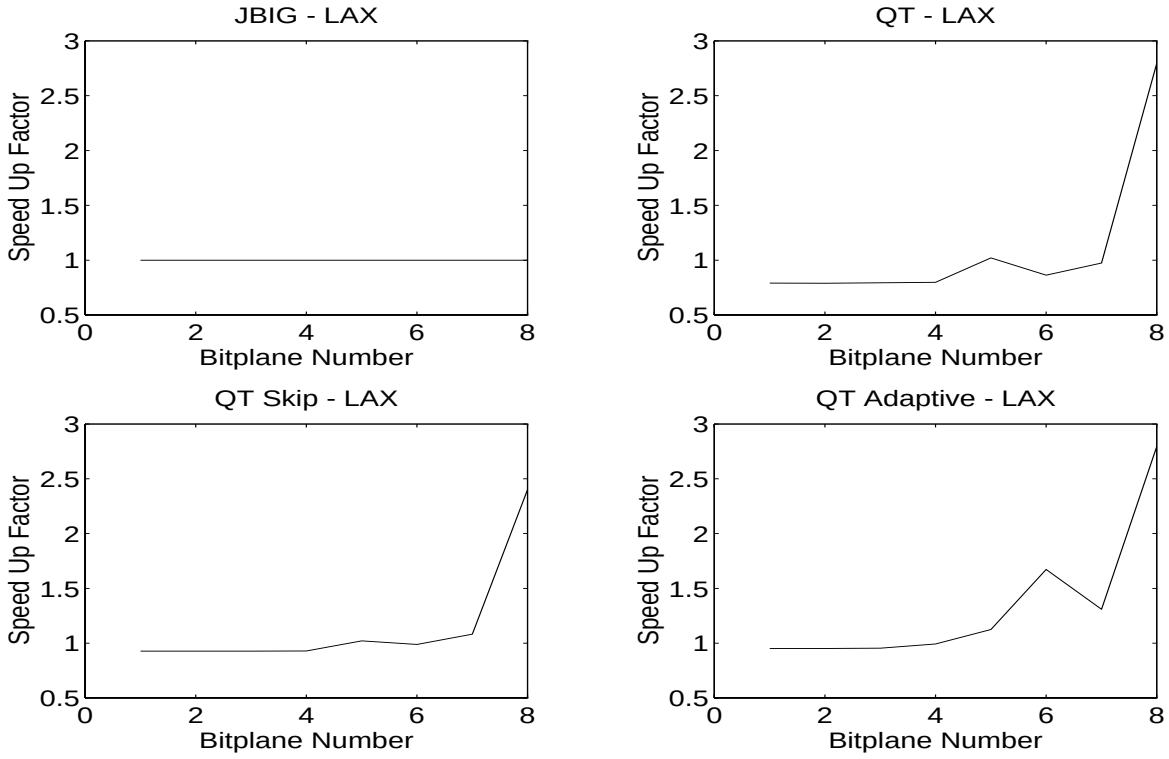


Figure 9.24: The image LAX is PCM gray coded, the relative speed of each algorithm is shown

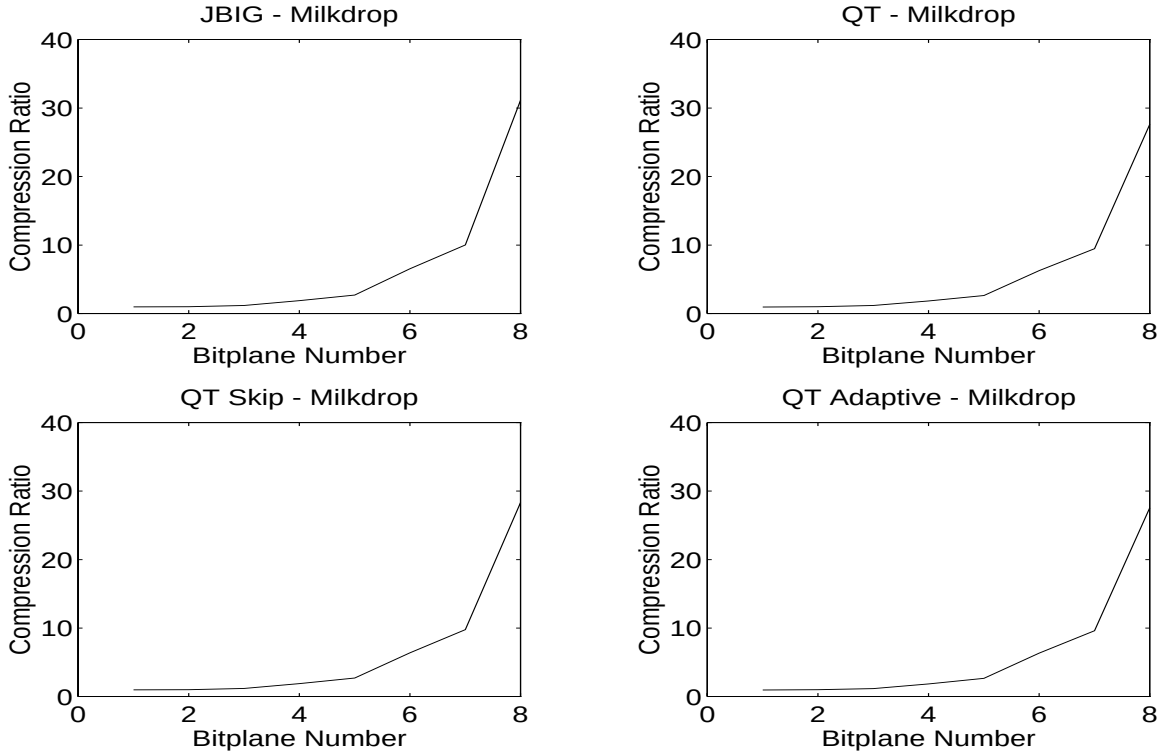


Figure 9.25: The image Milkdrop is PCM gray coded, the compression ratio of each algorithm is shown

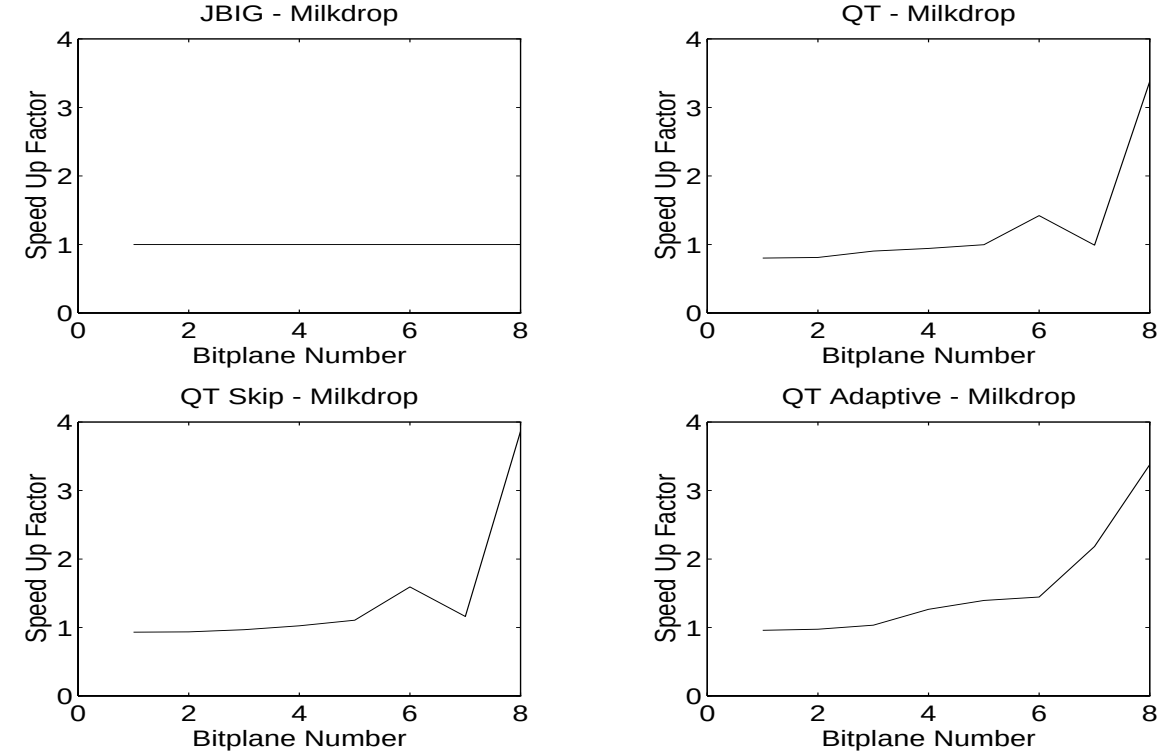


Figure 9.26: The image Milkdrop is PCM gray coded, the relative speed of each algorithm is shown

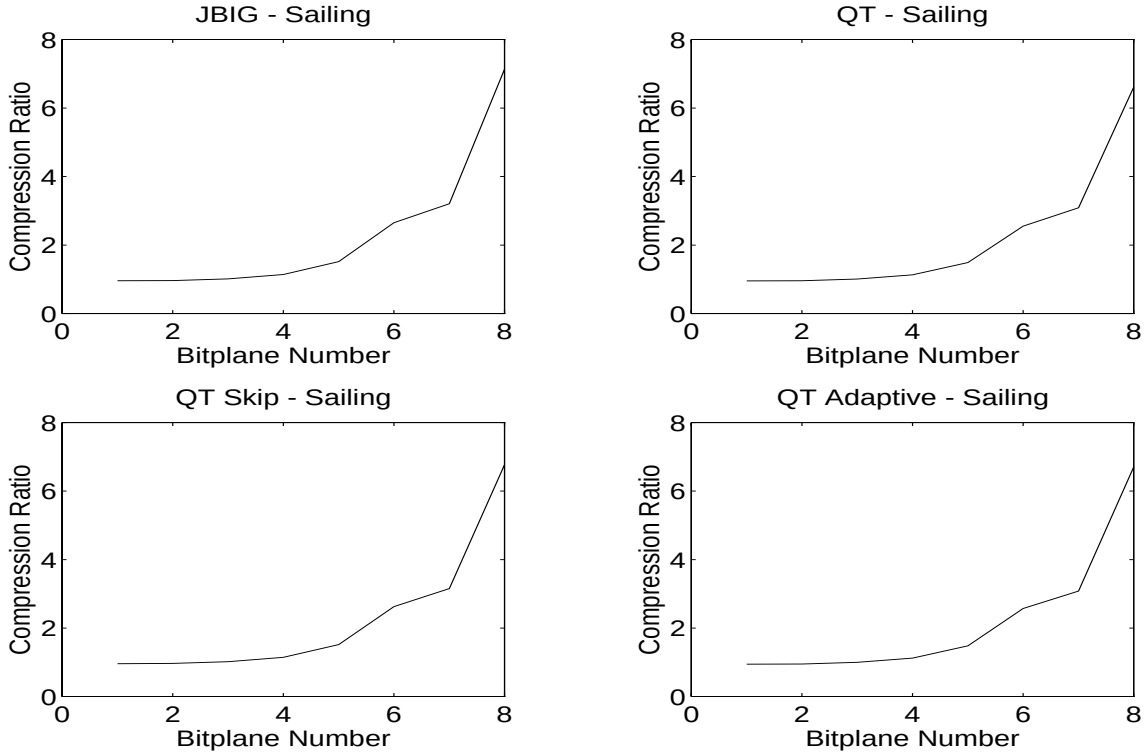


Figure 9.27: The image Sailing is PCM gray coded, the compression ratio of each algorithm is shown

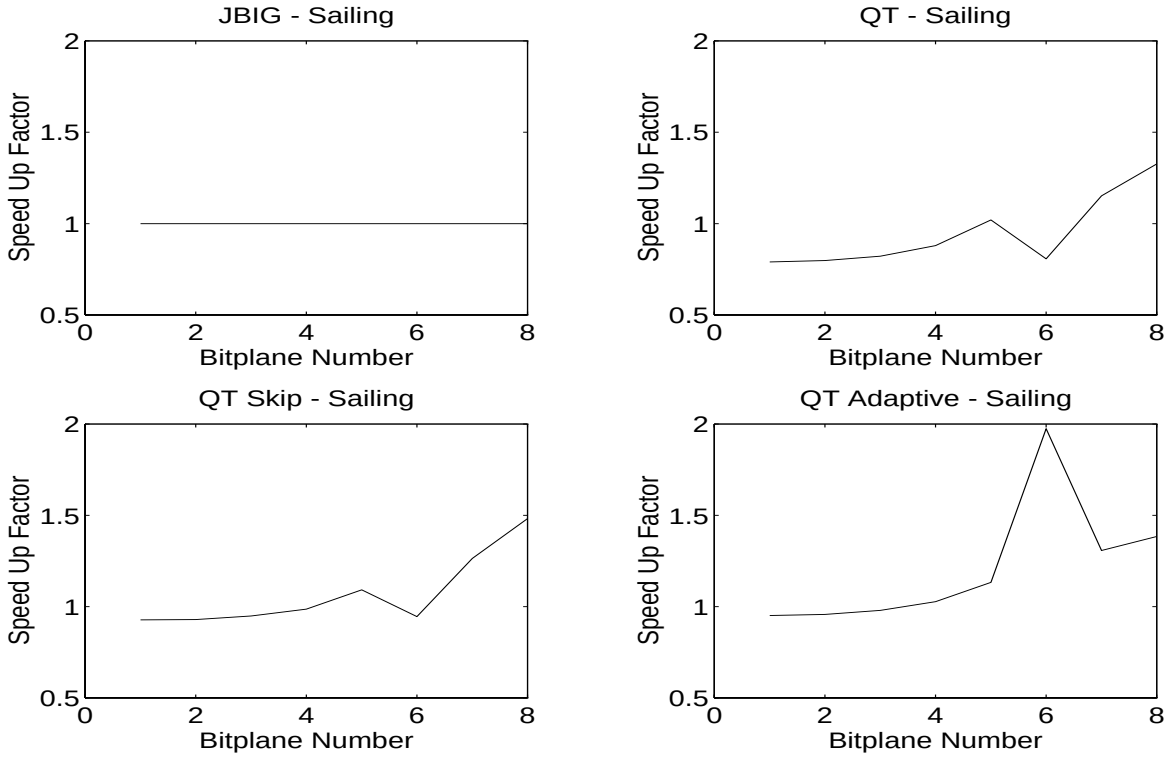


Figure 9.28: The image Sailing is PCM gray coded, the relative speed of each algorithm is shown

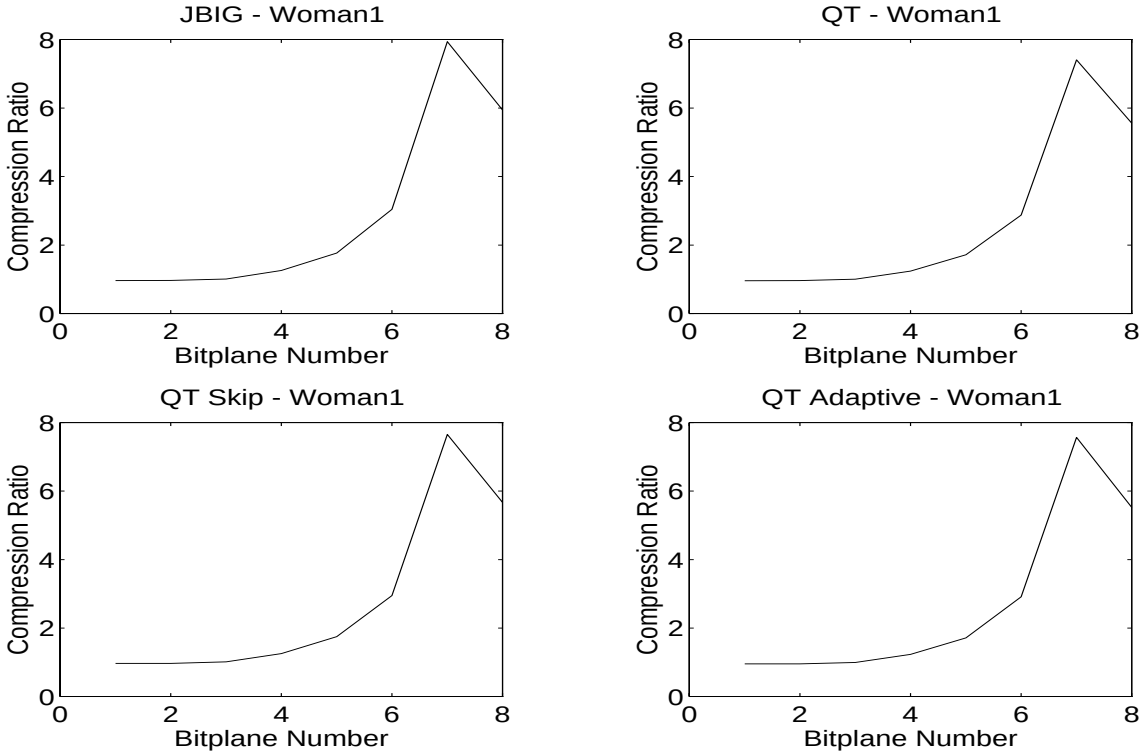


Figure 9.29: The image Woman1 is PCM gray coded, the compression ratio of each algorithm is shown

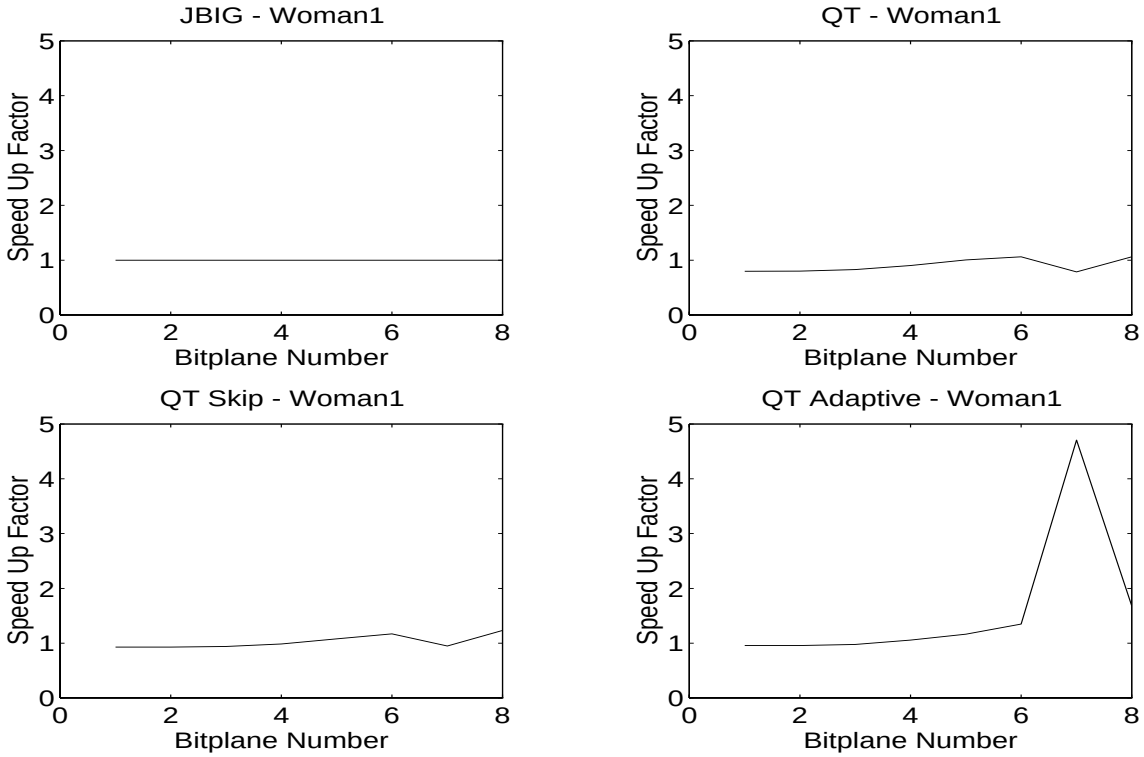


Figure 9.30: The image Woman1 is PCM gray coded, the relative speed of each algorithm is shown

4 Bit PCM Data					
Algorithm	Image	Coded Bytes	Coded Ratio	Coded Bits	Speed
JBIG	LAX	64255	2.04	1048576	1
JBIG QT	LAX	65746	1.99	923316	1.14
QT Skip	LAX	64260	2.04	873536	1.20
QT Adapt	LAX	65721	1.99	683808	1.53
JBIG	Milkdrop	21425	6.12	1048576	1
JBIG QT	Milkdrop	22287	5.88	789872	1.33
QT Skip	Milkdrop	21795	6.01	695124	1.51
QT Adapt	Milkdrop	22133	5.92	567124	1.85
JBIG	Sailing	48769	2.69	1048576	1
JBIG QT	Sailing	49278	2.66	1006816	1.04
QT Skip	Sailing	49278	2.66	901548	1.16
QT Adapt	Sailing	50390	2.60	754008	1.39
JBIG	Woman1	38941	3.37	1048576	1
JBIG QT	Woman1	40823	3.21	1088884	0.96
QT Skip	Woman1	39901	3.28	956420	1.09
QT Adapt	Woman1	40644	3.22	630848	1.66

Table 9.7:

9.3.4 Conclusion

The quadtree algorithms achieved only 5-30% speedup at a cost of 2-10% in compression when compared with non-progressive JBIG on the sigma delta modulated and gray coded data used in this study. All of the algorithms achieved lower speed and compression ratios on the sigma delta modulated data than on the gray coded data. We also found that sigma delta modulated bit planes are easier to compress, i.e. are faster and achieve higher compression ratios, after the modulator is reset. This occurs because a one-bit first-order sigma delta modulator encodes the analog input value using not only the number of plus and minus one, but also the locations of the bits. This decorrelates adjacent pixels, and lowers the achievable compression. During normal operation we do not plan to reset the sigma delta modulators of our image sensor. Therefore, all of the quadtree algorithms discussed in this section will not effectively compress data generated by our image sensor. In order to compress sigma delta modulated data we believe that decimation must proceed any compression

algorithm.

Chapter 10

Conclusions

In this thesis we have presented the first known CMOS area image sensor with pixel A/D conversion. The A/D conversion is performed using a one-bit first-order sigma delta modulator at each pixel. The sensor outputs digital data and allows for both programmable quantization and region of interest windowing. We showed two separate circuit implementations of this sensor, and presented experimental results for two test chips. This demonstrated the viability of our image sensor approach, but it also helped us to understand its limitations. Although our sensor dissipates very little power < 50nW per pixel and has a wide dynamic range > 80dB, its application is limited by large pixel size and low fill factor. We believe that future designs can overcome these limitations with circuit ideas being developed in our laboratory.

Our sensor uses sigma delta modulation for A/D conversion. Therefore, we demonstrated several techniques for decimating both still and video image data. Nonlinear decimation techniques show great promise for still image data where real time operation is not essential. These techniques achieved 5-10dB higher average SNR than linear techniques. Since video decimation requires real time operation nonlinear decimation technique are not practical. Linear techniques were shown to be effective for video data, but at a cost of 20-30dB in average SNR. Decimation of video signals is hard, it requires high I/O data bandwidths and or large amounts of external memory.

In order to reduce the amount of the data transmitted from our image sensor, we investigated data compression of the sigma delta modulated bit planes. This proved

to be difficult and we showed that trying to compress the bit planes one at a time achieved an average compression ratio of 1.05. This poor compression was caused by adjacent sigma delta modulators decorrelating the original image data. To compress images from our sensor the data must be decimated first. Although, we found that sigma delta modulated bit planes are difficult to compress we did develop a new JBIG compliant algorithm that provides a 60% speed improvement over the fasted known JBIG algorithm, with a 2% increase in compression when tested on CCITT FAX documents.

10.1 Future Work

We will discuss three areas of future work for our image sensor. The first is improving the characteristics of our image sensor, such as resolution, pixel size, and fixed pattern noise. In Chapter 6 we described an image sensors with muxed pixel-level A/D conversion. In a 0.35μm CMOS process, this approach is expected to produce an image sensor with characteristics superior to CCDs. In addition to resolution, pixel size, and fixed pattern noise, future image sensors should provide image enhancement throught local and global signal processing. This can be used to correct imprefections in the camera system, and poor lighting.

The second area is integrating a decimation circuit and an image sensor on the same chip. Although, the preliminary work in this area was discussed in Chapters 7 and 8 more circuits and layout work are necessary for a complete implementation. The most difficult design constraint is imposed by the I/O bandwidth requirement. The I/O pads will need to operate at high frequencies, but they should also produce as little substrate noise, i.e. impact ionization current, as possible. One method that might achieve this goal is low swing I/O pads. This method was successfully used to minimize substrate noise by Loinaz in [42].

The final area is investigating lossy compression algorithms for sigma delta modulated data. In Chapter 9 we presented our work on lossless image compression of sigma delta modulated data. We also concluded that these algorithms are ineffective. Therefore, lossy compression techniques such as [28, 50, 22, 57, 27] are the next step

in the search for a viable on-chip compression algorithm.

Appendix A

CMOS Scaling Issues

This appendix describes some of the CMOS process scaling limitations imposed on image sensors with integrated A/D conversion. We assume throughout this appendix that the size of the image sensor die is constant and that the number pixels increases as the square of the process shrink.

In order to quantify the effect of process scaling we will use an analysis method similar to Burr [10]. This method uses a generic scaling parameter S, to determine system performance. S is defined as a ratio of minimum channel lengths, i.e.

$$S = \frac{\text{Present minimum channel length}}{\text{scaled minimum channel length}}. \tag{A.1}$$

The assumptions used to related device parameters to S are shown in Table A.1.

A CMOS image sensor with a single A/D converter does not scale well. Since the number of pixels in the image sensor will increase as S^2 because of horizontal and vertical process scaling, the A/D converter will have to operate S^2 times faster to keep up with the data. The process induced speed up for submicron devices is limited to $S^{1.5}(1 - \frac{V_{th}}{V_{dd}})$ because of velocity saturation. The only way the A/D converter can keep up with the increased data requirement is to increase the A/D converters size, i.e. use pipelined and or parallel circuitry. This is not a desirable design trade-off.

A CMOS image sensor with a semi-parallel A/D converter scales much better than a CMOS image sensor with one A/D converter, but this topology also has its

CMOS Technology Scaling		
Parameter	Scaling	Description
Tech	S	$\frac{\lambda_0}{\lambda}$
X_l	$\frac{1}{S}$	Horizontal Dimensions
Z_l	$\frac{1}{S}$	Vertical Dimensions
T_{ox}	$\frac{1}{S}$	Gate Oxide
C_{ox}	S	Gate Capacitance
U_0	$S^{-0.5}$	Carrier Mobility
$C_{ox}U_0$	$S^{0.5}$	Transconductance
C_j	$\frac{S}{V^{0.5}}$	Diffusion Capacitance
C_{sjw}	S	Sidewall Capacitance
I_{ds}	$S^{0.5}(V_{dd} - V_{th})$	Velocity Saturated Drain Current
Q	$\frac{V_{dd}}{S}$	Device Charge
$\frac{I_{ds}}{Q}$	$S^{1.5}(1 - \frac{V_{th}}{V_{dd}})$	Process Performance

Figure A.1: CMOS Technology Scaling

limitations. Since the A/D conversion is distributed over a large number of simple converters, usually one converter for each column of the image sensor array, the speed of the A/D converter array will scale as fast as the size of future image sensors. Clearly, the number of rows in an image sensor will increase by S, and the speed of each A/D converter will increase by $S^{1.5}(1-\frac{V_{th}}{V_{dd}})$. Therefore, the speed of semi-parallel A/D converter topologies should scale with CMOS technology. These sensor are not flawless, they have analog communication limitations between the sensor core and the semi-parallel A/D converter array. As the size of the sensor increases it will become very difficult to quickly and accurately communicate the pixel values to the edge of the image sensor array. The pixel sense line capacitance should scale as 1, because the number of pixels in the horizontal direction scales as S and the effective capacitance of each pixel scales as S^{-1}. I_{ds} scales as $S^{0.5}(V_{dd} - V_{th})$ for velocity saturated submicron MOS transistors, the bandwidth per pixel scales as S^{-1}. Therefore, the pixel sense line bandwidth per pixel scales as $S^{-0.5}(1 - \frac{V_{th}}{V_{dd}})$.

Finally, CMOS image sensors with parallel A/D conversion scale well with technology. Since each pixel or each group of pixels has its own A/D converter the A/D

converters speed can remain constant as technology shrinks. Parallel A/D conversion does have the same communication problems associated with the semi-parallel topologies, with the exception that the data being communicated is digital. Therefore, it is simpler to detect the data, and we can make better use of the bit line bandwidth. We also benefit from advances in DRAM sensing technology.

Bibliography

[1] I. Akiyama et al. "A 1280 x 980 Pixel CCD Image Sensor". In *ISSCC Digest of Technical Papers*, San Fransico, CA, February 1986.

[2] R.B. Arps, T.K. Truong, D.J. Lu, R.C. Pasco, and T.D. Friedman. A Multipurpose VLSI Chip for Adaptive Data Compression of Bilevel Images. *IBM J. of Research and Development*, 32(6), November 1988.

[3] Anders Astrom. *Smart Image Sensors*. PhD thesis, Lingoping University, 1993.

[4] M.S. Bazaraa, J.J. Jarvis, and H.D. Sherali. *LINEAR PROGRAMMING AND NETWORK FLOWS*. Wiley, 1990.

[5] W.R. Bennett. Spectra of quantized signals. *Bell Systems Technical Journal*, 27:446–472, July 1948.

[6] B. E. Boser et al. Simulating and Testing Oversampled Analog-to-Digital Converters. *IEEE Transactions on Computer-Aided Design*, 7:668–674, June 1988.

[7] J. T. Bosiers et al. An S-VHS Compatible 1/3" Color FT-CCD Imager with Low Dark Current by Surface Pinning. *IEEE Transactions on Electron Devices*, 42(8):1449–1460, August 1995.

[8] R.W. Boyd. *RADIOMETRY AND THE DETECTION OF OPTICAL RADIATION*. Wiley-Interscience, 1983.

[9] R. N. Bracewell. *The Fourier Transform and Its Applications*. McGraw Hill, 1986.

[10] James B. Burr. Digital Neurochip Design. In K. Wojtek Przytula and Viktor K. Prasanna, editors, *Digital Parallel Implementations of Neural Networks*, pages 223–281. Prentice Hall, 1992.

[11] J. C. Candy. A Use of Double Integration in Sigma Delta Modulation. *IEEE Trans. Comm.*, 33(3):249–258, March 1985.

[12] J.C. Candy and G.C. Temes, editors. *Oversampled Delta-Sigma Data Converters.* IEEE Press, 1992.

[13] J.C. Candy, B. A. Wooley, and O.J. Benjamin. A Voiceband Codec with Digital Filtering. *IEEE Transactions on Communication*, 29:815–830, June 1981.

[14] S. Chu and C.S. Burrus. Multirate Filter Designs Using Comb Filters. *IEEE Transactions on Circuits and Systems*, 31:913–924, November 1984.

[15] R.E. Crochiere and L.R. Rabiner. Interpolation and Decimation of Digital Signals - A Tutorial Review. *Proceeding of the IEEE*, 69:300–331, March 1981.

[16] C.C. Cutler. Transmission systems employing quantization. U.S Patent No. 2,927,962, 1960. Filed 1954.

[17] Tobias Delbruck. *Investigations of Analog VLSI Visual Transduction and Motion Processing.* PhD thesis, California Institute of Technology, 1993.

[18] A. Dickinson, S. Mendis, D. Inglis, K. Azadet, and E. Fossum. CMOS Digital Camera With Parallel Analog-to-Digital Conversion Architecture. In *1995 IEEE Workshop on Charge Coupled Devices and Advanced Image Sensors*, April 1995.

[19] E.Fossum et al. A 256×256 Active Pixel Image Sensor with Motion Detection. In *ISSCC95 Techincal Digest*, February 1995.

[20] EG&G Reticon, 345 Potrero Ave., Sunnyvale, CA 94086-4197 USA. *Image Sensing Products 1992/1993*, 1992.

[21] EG&G Reticon. Image Sensing Products Domestic Price List, May 1993.

[22] P. Filip. An efficient method for image compression in the wavelet transform domain. In *Proceeding of the SPIE*, San Diego, CA, USA, July 1993. vol.2028, p. 72-81.

[23] R. Forchheimer and A. Odmark. A Single Chip Linear Array Processor, in Applications of digital image processing. *SPIE*, 1983.

[24] R. Forchheimer, P.Ingelhag, and C. Jansson. MAPP2200, a second generation smart optical sensor. *SPIE*, 1992.

[25] E.R. Fossum. Active Pixel Sensors: are CCD's dinosaurs. In *Proceeding of SPIE*, pages 2–14, San Jose, CA, February 1993.

[26] B. Fowler, A. El Gamal, and D. X. D. Yang. "A CMOS Area Image Sensor with Pixel-Level A/D Conversion". In *ISSCC Digest of Technical Papers*, San Fransico, CA, February 1994.

[27] Didier Le Gall. MPEG A Video Compression Standard for Multimedia Applications. *Communications of The ACM*, 34(4):47–58, April 1991.

[28] A. Gersho and R. M. Gray. *Vector quantization and signal compression.* Kluwer Academic Publishers, 1992.

[29] G.Feygin, P.G. Gulak, and P.Chow. Architectural Advances in the VLSI Implementation of Arithmetic Coders for Binary Image Compression. In *Proc. 1994 Data Compression Conference*, Snowbird, Utah, March 1994.

[30] L.A. Glasser and D.W. Dobberpuhl. *Design and analysis of VLSI circuits.* Addison Wesley, 1985.

[31] D.J. Goodman and M.J. Carey. Nine Digital Filters for Decimation and Interpolation. *IEEE Transactions on Acoustics, Speech, Signal Processing*, 25:121–126, April 1977.

[32] M. Gottardi, A. Sartori, and A. Simoni. POLIFEMO: An Addressable CMOS 128×128 - Pixel Image Sensor with Digital Interface. Technical report, Istituto Per La Ricerca Scientifica e Tecnologica, 1993.

[33] R. M. Gray. Oversampled Sigma-Delta Modulation. *IEEE Trans. Comm.*, 35(5):481–489, May 1987.

[34] R. M. Gray. Quantization Noise Spectra. *IEEE Transactions on Information Theory*, 36:1220–1244, November 1990.

[35] R. Gregorian and G.C. Temes. *Analog MOS integrated circuits for signal processing*. Wiley, 1986.

[36] S. Hein and A. Zakhor. *Sigma Delta Modulators: Nonlinear Decoding Algorithms and Stability Analysis*. Kluwer Academic Publishers, 1993.

[37] D. Hertog. *Interior point approach to linear, quadratic, and convex programming: algorithms and complexity*. Kluwer Academic Publishers, 1994.

[38] Information Systems Laboratory, Electrical Engineering Department, Stanford University, Stanford, CA 94305-4055. *Protozone The PC-Based ASIC Design Frame Users Guide*, August 1990.

[39] International Telegraph and Telephone Consultative Committee (CCITT). Progressive Bi-level Image Compression, February 1993. Recommendation T.82.

[40] M. Ito et al. A 10 bit 20MS/s 3 V Supply CMOS A/D Converter. *IEEE JOURNAL OF SOLID STATE CIRCUITS*, 29(12):1531–1536, december 1994.

[41] C.L. Keast and C.G. Sodini. A CCD/CMOS-Based Imager with Integrated Focal Plane Signal Processing. *IEEE Journal of Solid State Circuits*, 28(4):431–437, April 1993.

[42] M. Loinaz and B.A. Wooley. A CMOS Multi-Channel IC For Pulse Timing Measurements with 1mV Sensitivity. In *ISSCC Digest of Technical Papers*, San Fransisco, CA, February 1995.

[43] J. L. McCreary. Matching Properties, and Voltage and Temperature Dependence of MOS Capacitors. *IEEE Journal of Solid State Circuits*, 16(6):608–616, December 1981.

[44] C. Mead. "A Sensitive Electronic Photoreceptor". In *1985 Chapel Hill Conference on VLSI*, Chapel Hill, NC, 1985.

[45] C. Mead. *Analog VLSI and Neural Systems.* Addison Wesley, 1989.

[46] S. Mendis, E. Fossum, et al. A 128×128 CMOS Active Pixel for Highly Integrated Imaging Systems. In *IEEE IEDM Technical Digest*, San Jose, CA, December 1993.

[47] G. J. Michon and H. K. Burke. "Charge Injection Imaging". In *ISSCC Digest of Technical Papers*, pages 138–139, February 1973.

[48] R.S. Muller and T.I. Kamins. *DEVICE ELECTRONICS FOR INTEGRATED CIRCUITS.* Wiley, 1986.

[49] D.C. Youla nd H. Webb. Image Restoration by the Method of Convex Projections: Part 1- Theory. *IEEE Transactions on Medical Imaging*, MI-1:81–94, October 1982.

[50] A. N. Netravali and B. G. Haskell. *Digital Pictures Representation and Compression.* Plenum Press, 1988.

[51] T. Nobusada et al. Frame Interline CCD Sensor for HDTV Camera. In *ISSCC Digest of Technical Papers*, San Francisco, CA, USA, February 1989.

[52] A.V. Oppenheim and R.W. Schafer. *Digital Signal Processing.* Prentice Hall, 1975.

[53] A. Papoulis. *Probability Random Variables and Stochastic Processes.* McGraw Hill, 1984.

[54] P.Denyer, D. Renshaw, G. Wang, and M. Lu. A Single-Chip Video Camera with On-Chip Automatic Exposure Control. In *ISIC-91*, 1991.

[55] P.Denyer, D Renshaw, G. Wang, and M. Lu. CMOS Image Sensors For Multimedia Applications. In *CICC93*, 1993.

[56] P.Denyer, D. Renshaw, G. Wang, M. Lu, and S.Anderson. On-Chip CMOS Sensors for VLSI Imaging Systems. In *VLSI-91*, 1991.

[57] W.B. Pennebaker and J.L. Mitchell. *JPEG Still Image Data Compression Standard*. Van Nostrand Reinhold, New York, 1993.

[58] W.B. Pennebaker, J.L. Mitchell, G.G. Langdon, and R.B. Arps. An Overview of the Basic Principles of the Q-Coder Adaptive Binary Arithmetic Coder. *IBM J. of Research and Development*, 32(6), November 1988.

[59] T. Saramaki and H. Tenhunen. Efficient VLSI-Realizable Decimators for Sigma-Delta Analog-to-Digital Converters. In *IEEE Proceedings of ISCAS88*, pages 1525–1528, June 1988.

[60] R. Sarpeshkar, T. Delbruck, and C. A. Mead. White Noise in MOS Transistors and Resistors. *IEEE Circuits & Devices*, November 1993.

[61] Sheinwald et al. Deterministic prediction in progressive coding. *IEEE Transactions on Information Theory*, 39(2), March 1993.

[62] R. Steele. *Delta Modulation Systems*. Wiley, 1975.

[63] N. Tanaka et al. "A 310k Pixel Bipolar Imager (BASIS)". In *ISSCC Digest of Technical Papers*, San Fransisco, CA, February 1989.

[64] N. Tanaka, T. Ohmi, and Y. Nakamura. "A Novel Bipolar Imaging Device with Self-Noise Reduction Capability". *IEEE Trans. Elec. Dev.*, 36(1):31–38, January 1989.

[65] T.Bernard et al. A Programmable Artificial Retina. *IEEE Journal of Solid State Circuits*, 28(7), 1993.

[66] N.T. Thao and M. Vetterli. Oversampled A/D conversion using alternate projections. In *Twenty-Fifth Annual Conference on Information Sciences and Systems*, pages 241–248, 1991.

[67] T.Markas et al. Quad tree structures for image compression applications. *INFORMATION PROCESSING AND MANAGEMENT*, 28(6), 1992.

[68] M. Tremblay, D. Larendeau, and D. Poussart. Multi Module Focal Plane Processing Sensor with Parallel Analog Support for Computer Vision. In *Proceedings of the Symposium on Integrated Systems*, 1993.

[69] H. Tseng, B.T. Nguyen, and M. Fattahi. A high performance 1200*400 CCD array for astronomy applications. In *Proceeding of SPIE*, pages 170–185, 1989.

[70] E. A. Vittoz. The Design of High-Performanc Analog Circuits on Digital CMOS Chips. *IEEE Journal of Solid State Circuits*, 20(3):657–665, June 1985.

[71] R.M. Wehr, J.A. Richards, and T.W. Adair. *PHYSICS OF THE ATOM*. Addison Wesley, 1984.

[72] W. Yang. A Wide-Dynamic-Range Low-Power photosensor Array. In *ISSCC Digest of Technical Papers*, San Francisco, CA, USA, February 1994.