A Low Power Display Driver with Simultaneous Image Transformation

by

Jeremy Zaks Walker

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2005

Author .....
Department of Electrical Engineering and Computer Science
January 28, 2005

Certified by .....
Akintunde Ibitayo Akinwande
Professor
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

A Low Power Display Driver with Simultaneous Image Transformation

by

Jeremy Zaks Walker

Submitted to the Department of Electrical Engineering and Computer Science
on January 28, 2005, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

In this thesis, I designed, implemented, and evaluated the energy consumption of a system that uses a liquid crystal display to perform a one-dimensional transform. The RMS response of the liquid crystal elements themselves were exploited to perform a matrix multiplication (image transformation) over a single frame period. This image transformation was used as the last step of the decompression process in an image processing system. The system was implemented first in Matlab, then as a printed circuit board, and finally as an integrated circuit. While the initial Matlab and printed circuit board implementations looked promising, a number of practical considerations arose during the integrated circuit design that ultimately resulted in moderate performance: 14.3% energy savings.

Thesis Supervisor: Akintunde Ibitayo Akinwande
Title: Professor

Acknowledgments

I would first like to thank Professor Akintunde Ibitayo Akinwande for taking me on as his student and giving me the opportunity to work on this exciting project. I would like to sincerely thank him for his consistent support and encouragement throughout the research for and the development of my thesis. I would also like to thank Professor Anantha Chandrakasan for the use of his lab as well as his assistance in this thesis; his comments were always insightful and direct. Thanks go also to Ben Calhoun, Nathan Ickes, and Frank Honore for all of their help in the lab and with my circuit design. Thanks are due to the support team at Austriamicrosystems. Without their assistance, much of this thesis would not be where it is now. I also thank Samuel Crooks, Professor Charlie Sodini and again Akintunde Ibitayo Akinwande. Without their knowledge of the particulars of Non-Disclosure Agreements, much of this thesis would not have been possible.

I can't thank my family enough for all of the support and encouragement they gave me throughout this process. Thanks to Jen, Paul, Ravi, Sean and Benjamin for a great year at 3 Leonard and thanks to Buddhika for a great last semester to finish it off. Thanks to Amy Seng for listening to me discuss my thesis night after night and thanks in particular to Denis Daly whose friendship, help, and encouragement continue to amaze me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Multimedia-enabled mobile devices have recently exploded into mainstream commercial use. Two major, and often conflicting, demands exist in this market: low power and high quality. High quality multimedia applications demand high resolution screens capable of displaying fast-moving images. Unfortunately, the higher the resolution and the rate of change of the images, the more energy is required to process and display them. Not only is more energy required, but higher bandwidth is also necessary since more data must be processed in a fixed amount of time.

While the mobile market is the first to be concerned with energy and bandwidth requirements, it is easy to see that as displays move to higher and higher resolutions, these concerns begin to affect all manner of displays. For example, currently a standard VGA screen has 640x480 pixels (307 200). At a refresh rate of 100Hz and 8-bits per pixel, a data rate of 250Mb/s is required. A higher definition screen that is 6400x4800 pixels at a refresh of rate 100Hz and 8-bits per pixel, a data rate of 25Gb/s would be required. This bandwidth requirement increases as N^2 where N is the number of pixels along either the vertical or horizontal dimensions.

Considering a 3-dimensional display, the requirements become even more extreme, increasing with N^3.

It is clear that techniques that can reduce energy/bandwidth and also scale with

the size of a display will be extremely useful in facilitating the development of high-resolution displays. This thesis attempts to address this issue.

Chapter 1 introduces the relevant technology as well as the technique by which energy reduction may be possible. Chapter 2 addresses some of the practical considerations of using the technique such as gray shade generation and the effectiveness of the technique at reducing energy. Chapters 3 and 4 describes the design process moving from software prototype, to various hardware implementations. Specifically, it describes the Matlab implementation, the printed circuit board (PCB) implementation and finally the design of an integrated circuit (IC) display driver. Chapter 5 draws conclusions from the work done as well as suggesting further improvements to the IC and future work that could be pursued in this area. Appendix A contains time-domain simulations of the D/A and analog buffer used in the IC.

1.2 Liquid Crystal Displays (LCDs)

LCDs have quickly become a popular display technology because of their small size, easy manufacture and relatively low energy consumption. They have made the most headway in the mobile market where these qualities are particularly valued. There are two types of LCD that exist in the market today: active matrix displays and passive matrix displays.

1.2.1 Active Matrix Displays

An N $\times$ N (N rows by N columns) active matrix display incorporates the use of an active element, often a transistor, and also allows for a memory of some sort at the pixel itself. Active matrix displays have pixel elements that consist of a pass transistor, hold capacitor and liquid crystal (LC) that can be modeled as a capacitor in parallel with the hold capacitor. The hold capacitor is used to maintain a voltage across the liquid crystal during the frame period while the pass transistor serves to isolate the pixel element from all other pixel elements in the same column. The row lines in this case are attached to the gates of the pass transistors while the column

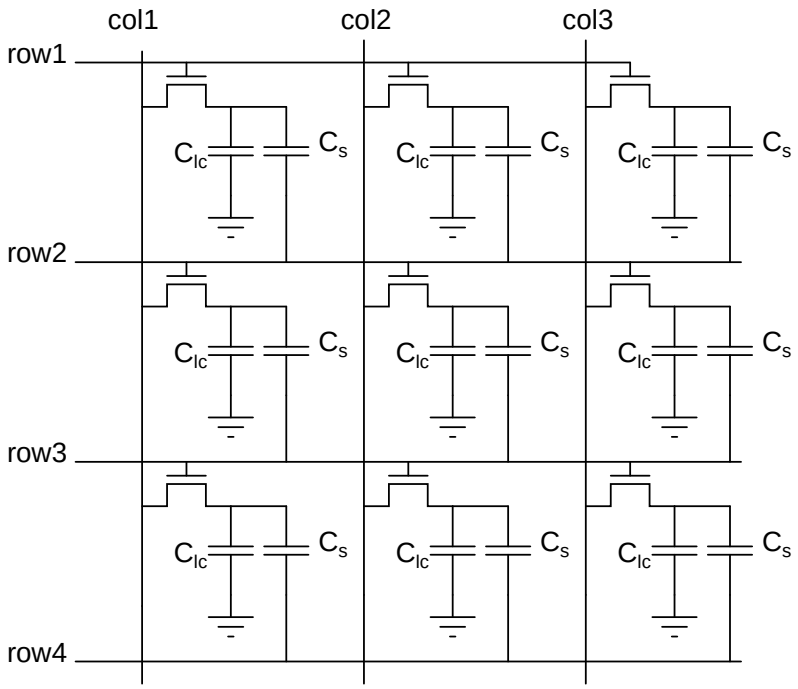

Figure 1-1: Active Matrix Display

lines are connected to one end of the pass transistor. When a row is selected, the pass transistor steers data from the column lines to the particular LC pixel.

1.2.2 Passive Matrix Displays

The standard N $\times$ N passive matrix display consists of N parallel row wires and N parallel column wires. The rows and the columns run perpendicular to each other in planes separated by some fixed distance. Between the wires the liquid crystal material is inserted, producing a row-liquid crystal-column sandwich. Where the row and column wires intersect, a pixel forms. This pixel can be controlled by modulating the voltage difference between the row and column wires.

In general, it has been shown that liquid crystals respond to both positive and negative voltages (ie. the pixel turns on when either a positive or negative voltage is applied) [12]. Given a step in voltage across the pixel, the change in light output is not instantaneous but rather takes a certain amount of time, generally a few ms. It has been shown that if a high enough frequency signal is applied across the liquid

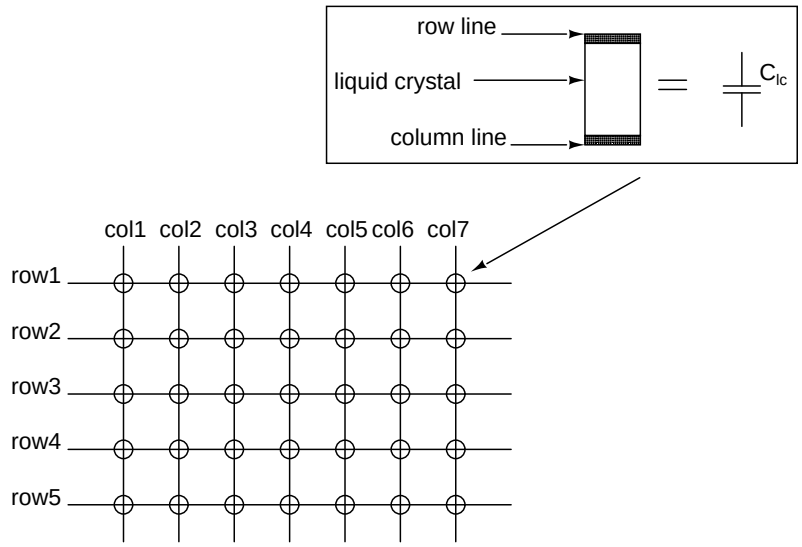

Figure 1-2: Passive Matrix Display

crystal (ie. a square-wave whose period is much less than the response time of the pixel), the light output through the pixel is a function of the RMS voltage across it [2].

For reasons that will be discussed further below, this thesis concerns itself entirely with passive matrix displays.

1.3 Addressing Techniques

Displays are generally addressed on a frame-by-frame basis where a single frame consists of all the pixel data required to display a static image. More completely, the data for one frame is placed into the display, then the data for the next frame is, and so on. Each frame is in turn composed of N row periods. During each row period, the data for that row is, in general, placed onto the column lines so that after N row periods, all N rows have all of their data. Currently, there are two main techniques to address a passive matrix display: row-at-a-time and multiple-line-addressing (MLA).

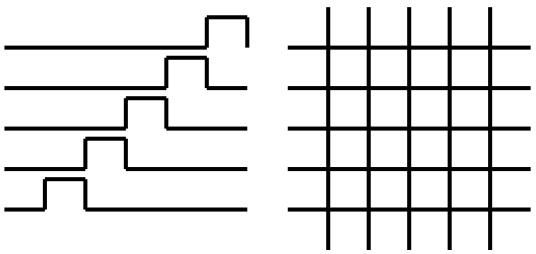

Figure 1-3: Row-at-a-time Addressing

1.3.1 Row-at-a-Time Addressing

Row-at-a-time addressing implies that one row is selected during a row period and all of the data for that row is placed on the columns. During the next row period, the next row is selected and the appropriate data is placed on the columns. As such, all pixels receive their values after N row periods. Since there is no hold capacitor as in the active matrix case, the row and column voltages must be carefully selected so that the RMS of the pixel voltages yields the correct voltage to turn the pixel on or off.

1.3.2 Multiple Line Addressing

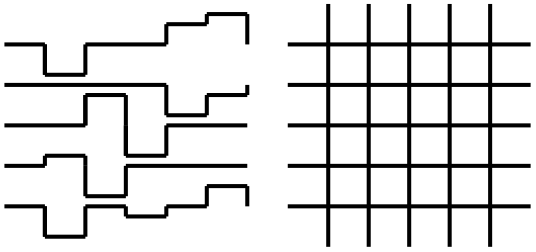

Figure 1-4: Multiple Line Addressing

MLA is an addressing technique in which instead of selecting a single row at a time, multiple rows are selected and some function of their data is placed on the columns.[1] This scheme was first implemented by Scheffer and Clifton [18]. Previous

[1]Ideas for this section were taken from Ernst Lueder, *Liquid Crystal Displays*, John Wiley & Sons Ltd., 2001

to them, Nehring and Kmetz [17] showed that given row and column voltage matrices, each pixel will display the correct value if the row functions (ie. the values applied to the rows over time) are orthonormal and the columns are computed as a function of these row voltages as well as the pixel information matrix, A. This is illustrated in more detail below.

Assume that the row voltages are given by $F_i(t)$ and the column voltages by $G_j(t)$ in an N $\times$ N display, where i corresponds to the row and j to the column in the N $\times$ N display matrix.

The column voltage waveforms must be calculated as a function of the row voltages and the pixel information matrix (a matrix of the image values) A:

$$G_j(t) = c \sum_{k=0}^{N-1} A_{kj} F_k(t)$$

where $F_i(t)$ are the row waveforms. This can also be expresed in matrix form as $G = cF^T A$.

To see more clearly how these conditions result in an undistorted image, we'll derive the pixel voltage after a single frame period below.

The voltage at a particular pixel at time t is simply the difference between the row and column voltages.

$$P_{ij} = F_i(t) - G_j(t)$$

Taken over a frame, the liquid crystal responds to the RMS voltage, given by

$$P_{ij_{rms}} = \sqrt{\frac{1}{N} \sum_{t=0}^{N-1} (F_i(t) - G_j(t))^2}$$

where N is the number of row periods in a single frame.

Expanding this

$$P_{ij_{rms}} = \sqrt{\frac{1}{N}\sum_{t=0}^{N-1}(F_i(t)^2 - 2F_i(t)G_j(t) + G_j(t)^2)}$$

If the rows signals are all orthonormal, then

$$\sum_{t=0}^{N-1} F_i(t)^2 = constant = NF^2$$

$$\sum_{t=0}^{N-1} F_i(t)F_k(t) = 0, i \neq k$$

Now, expanding $G_j(t)$,

$$G_j(t) = c\sum_{k=0}^{N-1} A_{kj}F_k(t)$$

The new expression for the pixel voltage is,

$$P_{ij_{rms}} = \sqrt{\frac{1}{N}\sum_{t=0}^{N-1}(F_i(t)^2 - 2cF_i(t)\sum_{k=0}^{N-1} A_{kj}F_k(t) + (c\sum_{k=0}^{N-1} A_{kj}F_k(t))^2)}$$

If all A_{ij} are +/- 1, this simplifies to,

$$P_{ij_{rms}} = \sqrt{\frac{1}{N}(NF^2 - 2cNF^2A_{ij} + c^2N^2F^2)}$$

which in turn simplifies to,

$$P_{ij_{rms}} = F\sqrt{1 - 2cA_{ij} + c^2N} \tag{1.1}$$

The constant c can be determined such that the on/off voltage ratio is maximized. Clearly each pixel value, $P_{ij_{rms}}$ is only a function of its data value A_{ij} and therefore each pixel displays the correct value.

There are a number of benefits of MLA [9], two of which are:

1. Decreased senstivity to fast liquid crystal response times,

2. Decreased energy consumption.

The first is apparent when one considers that if the liquid crystal response time is fast, then instead of responding perfectly to the RMS voltage, the liquid crystal may start to relax (ie. turn off) during the time when there is no voltage applied across it. This would cause flicker in the display. In the row-at-a-time addressing scheme, the liquid crystal only has sigificant voltage applied across it once per frame period, making the possibility for flicker very high. With MLA, on the other hand, the liquid crystal has voltage applied across it multiple times per frame and is therefore unlikely to have the opportunity to relax over the course of the frame period. The result is that faster-responding liquid crystals can be used, allowing for higher frame rates. This improved response to fast-responding liquid crystals was first demonstrated by Scheffer and Clifton [18].

Decreased energy consumption with the MLA scheme is not readily apparent but becomes so when one considers the required driving voltages to address a LCD. In the row-at-a-time scheme, the driving voltages must be high enough such that the RMS voltage across the liquid crystal is sufficient to turn the liquid crystal on, despite a long frame period. As frame periods remain fixed, in order to maintain a constant frame refresh rate, and resolutions increase, each pixel is driven for a decreasing percentage of the total frame period. Because of this, higher and higher driving voltages are required. If the driver circuits have any bias currents that are fixed, perhaps to bias a differential or gain stage, the higher supply voltage implies increased power, and therefore energy, consumption.

In a MLA display, since the pixel is being driven multiple times per frame period, and since the RMS voltage needs to have approximately the same value as the row-at-a-time scheme, the individual pixel voltages must be lower than the row-at-a-time scheme. These reduced voltages in turn allow for reduced energy consumption. The amount of reduction will depend heavily on the row waveforms used. This reduction in energy has been demonstrated in [3] and [25].

1.4 MLA and Matrix Multiplication

In the previous section, it was shown that even with multiple rows selected at once, each pixel still displays the correct value as long as the row matrix is orthonormal. This concept of the row and column waveforms as matrices is important since, as part of the RMS response of the display, a matrix multiplication of the row and column matrices is computed [11]. This multiplication is a by-product of the RMS response of the liquid crystal and is outlined here for clarity:

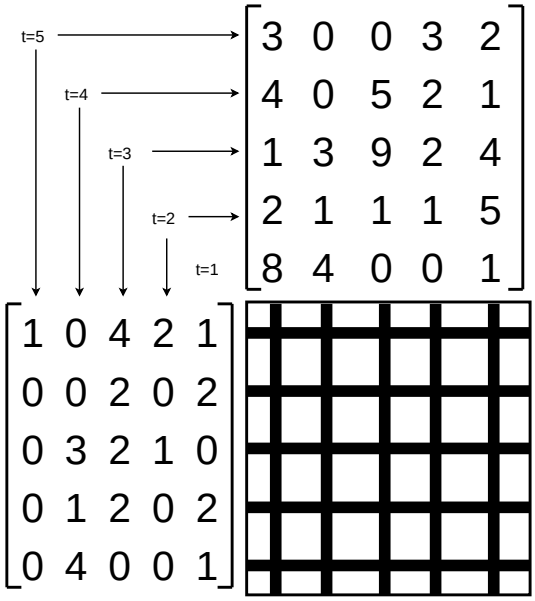

Figure 1-5: MLA as Matrix Multiplication

As seen in Figure 1-5, during each row period a series of column and row voltages is placed on the display. Each liquid crystal element, because of its RMS response, squares the difference of these voltages and adds it to its previous value. When all of the row periods have passed, taking the square root of the average of this sum yields the RMS value at the pixel. Now, during each row period, the liquid crystal squares the difference of the row ($F_i(t)$) and column ($G_j(t)$) voltages, yielding three terms.

$$(F_i(t) - G_j(t))^2 = F_i(t)^2 - 2F_i(t)G_j(t) + G_j(t)^2$$

The middle term corresponds to the multiplication of the row and column voltages while the other two are just the square of the same. By summing these terms over all row periods, three summation terms result.

$$\sum_{t=0}^{N-1} F_i(t)^2 - \sum_{t=0}^{N-1} 2F_i(t)G_j(t) + \sum_{t=0}^{N-1} G_j(t)^2$$

The middle term is proportional to the corresponding value of the matrix multiplication of the row and column matrices. If the rows are all normalized then the sum of their squares, the left term, will always be a constant, NF^2. The third term can also be made to sum to a constant, but this is explained later.

In sum, the display computes the square root of a single matrix multiplication with some constant offset terms.

1.5 MLA Matrix Multiplication as Image Transform

Any linear transform can be represented as a matrix multiplication. Because of this, it is possible to view the matrix multiplication that is happening in the liquid crystal display as being a 1-D transform. This transform is inherent to the liquid crystal display and therefore costs nothing in energy. Its specific use in saving energy is explained further in a later section.

1.6 Image Compression/Decompression

Image compression/decompression generally involves the following steps: transformation, quantisation, encoding, decoding, inverse quantization, and inverse transformation [4] [10]. Each step can be thought of as an operation performed on an original image that is represented by the matrix, A.

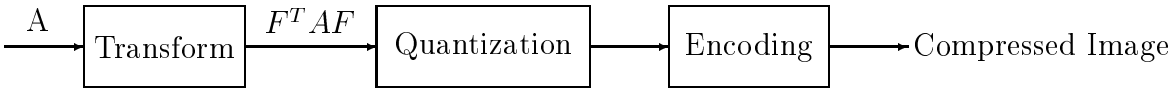

Figure 1-6: Generalized Image Compression

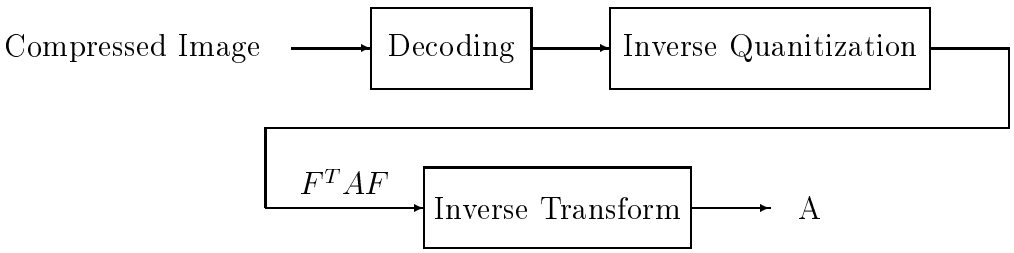

Figure 1-7: Generalized Image Decompression

Transformation of an image (a 2-D transformation) consists of two matrix multiplications each of which is a 1-D transformation. If A is the image matrix to be transformed and F is the transform matrix then B, the transformed image of A, is given by

$$B = F^T A F$$

After the image has been transformed, it is then encoded using any of a number of schemes. After encoding, it is in its most compact form and is ready for transmission through a channel to a receiver. The receiver receives the encoded data, performs decoding, and finally performs an inverse transform to recover the original image matrix, A.

1.6.1 Wavelet Transforms

In this thesis, wavelets are generally used to perform the image transform. Wavelets have proven to be particularly useful in image compression and have found adoption in the latest coding standards, such as JPEG2000 [7]. In general, wavelets are useful because they help to capture information in both the space and frequency domains. These wavelet transforms can be expressed in terms of matrix multiplications though it is also helpful to think of them as a filtering operation which produces some filtered version of the image.

In the 1-D case, the wavelet transform is often computed using a 2-channel filter bank as seen in Figure 1-8. The data is filtered using a pair of low-pass and high-pass filters. This segregation of low-pass and high-pass data yields higher scales of resolution. Essentially this means that more of the differences have been filtered out through the action of the low-pass filter. The resultant low-pass image constains less and less detail. Each high-pass output, on the other hand, represents some of the detail of the image. In order to maintain a constant amount of data (ie. constant number of samples), the outputs from the low-pass and high-pass filters are downsampled by a factor of 2.

The operation can be explained as follows:

In the first stage, the data is split into low-pass and high-pass bands. Subsequently, the data from the low-pass filter is then filtered using the same low-pass and high-pass filters, yielding subbands of the first low-pass band. This process of subsequent filtering is continued until either there is no more data to process (only a single sample is output from the previous low-pass filter) or the desired level of resolution is achieved.

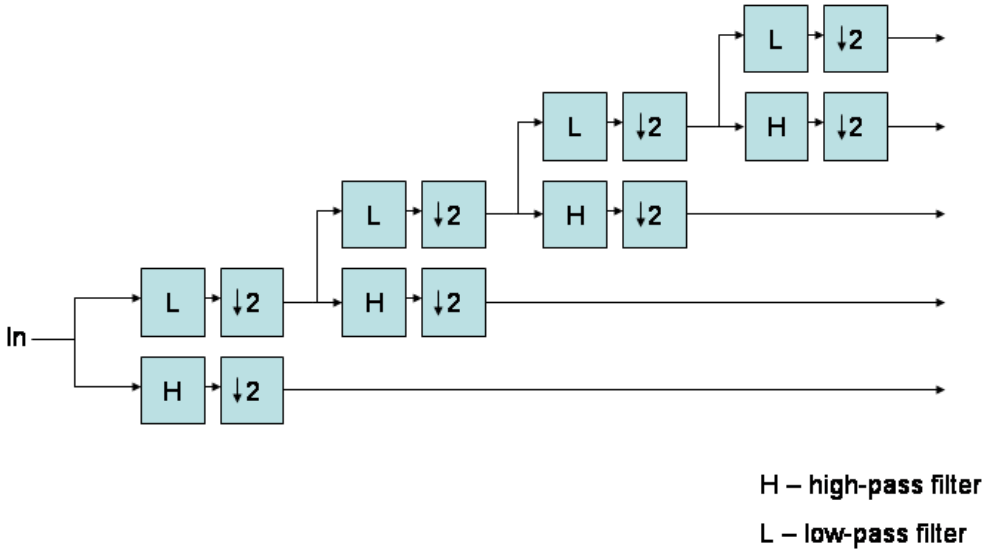

Figure 1-8: 1-D Transform as Mallat Decomposition

One way to perform a two-dimensional transform is to simply perform 2 1-D transforms on the image [26]. This is done by first filtering the data row-wise, and then filtering it column-wise for each scale. Since there are two outputs from the filter bank (low-pass, high-pass) and two dimensions (row, column), there will be four total outputs, each depending on whether the rows or columns were low-pass or high-pass filtered. This process is illustrated in Figure 1-9

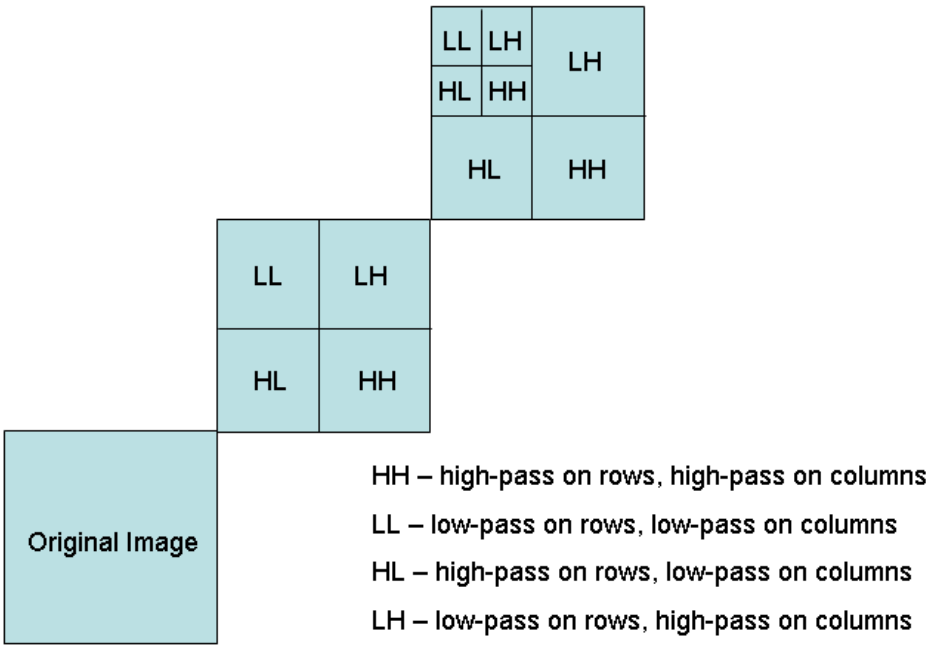

Figure 1-9: 2-D Transform

Subsequent scales of resolution are achieved by taking the LL output and passing it through the filter bank, yielding 4 outputs for the single input.

Wavelet Transform Matrix Construction

Filter banks are one way to visualize the filtering operations that are necessary to compute a wavelet transform[2] . However, these filtering operations can also be expressed as matrices and the result of their operation computed using a matrix multiplication. Let's look at a very simple wavelet such as the Haar Wavelet.

The coefficients of the low-pass filter are, $L = [\ 1 \quad 1\ ]$

The coefficients of the high-pass filter are, $H = [\ 1 \quad -1\ ]$

Let's assume that we have an input image, A, that is 4×4 in size

$$A = \begin{bmatrix} 2 & 3 & 3 & -2 \\ 1 & 0 & 2 & -3 \\ -2 & 1 & -1 & 0 \\ 1 & -3 & 2 & 1 \end{bmatrix} = \begin{bmatrix} A_{upper} \\ A_{lower} \end{bmatrix}$$

In order to perform the first set of operations (ie. low-pass and high-pass filtering followed by downsampling by a factor of 2), as shown in Figure 1-8, we can construct a matrix as follows:

$$F = \begin{bmatrix} L & 0 \\ 0 & L \\ H & 0 \\ 0 & H \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

By multiplying the two together we obtain:

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} 2 & 3 & 3 & -2 \\ 1 & 0 & 2 & -3 \\ -2 & 1 & -1 & 0 \\ 1 & -3 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 3 & 3 & 5 & -5 \\ -1 & -2 & 1 & 1 \\ 1 & 3 & 1 & 1 \\ -3 & 4 & -3 & -1 \end{bmatrix}$$

[2]Ideas for this section were taken from Gilbert Strang and Truong Nguyen, *Wavelets and Filter Banks*, Wellesley-Cambridge Press, Wellesley, Massachusetts, 1996

Symbolically,

$$\begin{bmatrix} L & 0 \\ 0 & L \\ H & 0 \\ 0 & H \end{bmatrix} \begin{bmatrix} A_{upper} \\ A_{lower} \end{bmatrix} = \begin{bmatrix} L(A_{upper}) \\ L(A_{lower}) \\ H(A_{upper}) \\ H(A_{lower}) \end{bmatrix}$$

By going through the multiplication step-by-step, it becomes clear that the top rows of the result matrix are the output of the low-pass filtering operation while the bottom rows are the output of the high-pass filtering operation.

The downsampling has also been taken care of in the matrix multiplication. By shifting the L operation to the right by two elements in the filtering matrix, one of the normally-computed low-pass outputs has been skipped over, precisely what would happen if it were first filtered and then downsampled. This can be more easily illustrated via an example where just the filtering operations but no downsampling is performed. Now, in order to compute just the filtering operations, a matrix such as the following would be required:

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix} = \begin{bmatrix} L & 0 & 0 \\ 0 & L & 0 \\ 0 & 0 & L \\ H & 0 & 0 \\ 0 & H & 0 \\ 0 & 0 & H \end{bmatrix}$$

Because each filtering operation (L,H) is shifted to the right by only one element in each successive row, all filter outputs will be computed. This makes sense since each column will now be processed as if it were a single incoming stream of data and after each time step (each successive row in the filter matrix) another filter output is computed.

To proceed with the example, multiplying this with the image matrix yields the

following:

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} 2 & 3 & 3 & -2 \\ 1 & 0 & 2 & -3 \\ -2 & 1 & -1 & 0 \\ 1 & -3 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 3 & 3 & 5 & -5 \\ \mathbf{-1} & \mathbf{1} & \mathbf{1} & \mathbf{-3} \\ -1 & -2 & 1 & 1 \\ 1 & 3 & 1 & 1 \\ \mathbf{3} & \mathbf{-1} & \mathbf{3} & \mathbf{-3} \\ -3 & 4 & -3 & -1 \end{bmatrix}$$

Subsequent downsampling by a factor of two would remove the 2nd and 5th rows (boldface type) and leave the matrix we had computed originally. It is important to note that the downsampling occurs on the outputs of the low-pass and high-pass filters independently; i.e., downsampling is started at the beginning of each filter's output so that only the second row in each filter's output is removed. These rows correspond to the 2nd and 5th rows in the matrix, respectively.

So far only the first scale of resolution (ie. the output of the first set of filter banks) has been computed. In order to compute multiple scales of resolution, a number of matrix multiplications can be cascaded together. For example, the first matrix, F, performs the original operations of high-pass and low-pass filtering (along with associated downsampling). In order to perform the next level of filtering operations it is necessary to operate only on the output of the first low-pass filter and to simply pass the output of the first high-pass filter. This can be accomplished with the following matrix, F_2:

$$F_2 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} L & 0 \\ H & 0 \\ 0 & I \end{bmatrix}$$

where

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

In order to perform multiple scales of resolution, it is necessary to first multiply the image by F, and then to multiply it by F_2. However, given the knowledge that multiple scales of resolution are desired, a single matrix can be constructed by multiplying together the F and F_2 matrices. Because of this, only a single matrix multiplication need be performed in order to obtain multiple scales of resolution.

All of the above examples were 1-D transforms (ie. each matrix multiplication resulted in filtering along the columns of the image matrix). In order to perform a full 2-D image transform, it would be necessary to perform filtering along the rows as well. This can be accomplished by post-multiplying the image matrix by F^T and $(F_2)^T$. The transpose operation flips the rows and columns so that the operations are performed along the rows instead of the columns. In other words, to perform the full 2-scale, 2-D transform, the following would need to be computed:

$$F_2 F A F^T (F_2)^T$$

1.7 Image Decompression and MLA

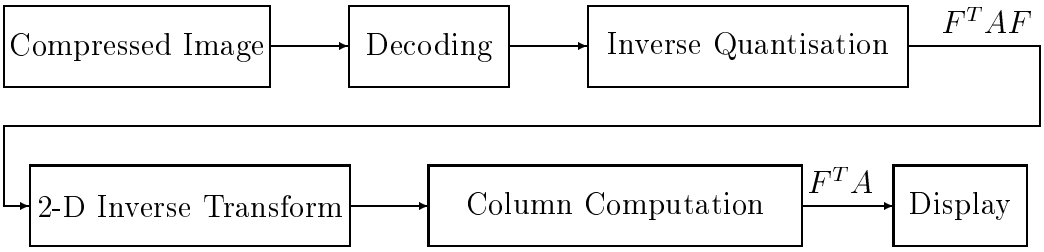

Figure 1-10: Normal Image Decompression + Display Driving[10]

It has been shown that a LCD naturally computes a matrix multiplication of its row and column matrices. This multiplication can be taken advantage of when considered in the context of image decompression as outlined above[10]. In particular, if the same matrix that is used to transform the image is used to drive the display and this matrix is orthogonal, both inverse transformation and the driving of the display can be accomplished simultaneously [10]. Consider the following example:

In a normal MLA display system and an orthonormal matrix F, given the transformed image matrix, $B = F^T AF$, the processor first post-multiplies B by F^T, giving

$$BF^T = F^T AFF^T = F^T A$$

It then pre-multiplies the result by F, giving

$$FF^T A = A$$

Since A is the pixel information matrix, this is then transmitted to the display where, as shown above, the column signals are computed as

$$G = cF^T A$$

Now, since part of the inverse transform already results in $F^T A$ which are the column signals needed by an MLA system, it makes sense to combine the inverse transform and column computation steps together in order to reduce the total number of operations.

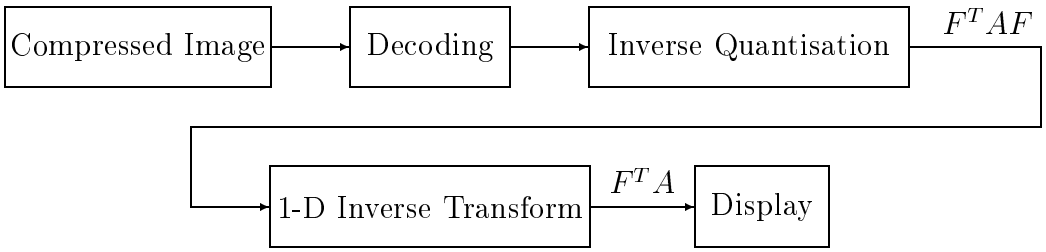

Figure 1-11: Image Decompression with Integrated Display Driving[10]

It is clear that by using an appropriate transform/driving matrix, both display driving and inverse transformation can be accomplished at the same time. This can, in turn, potentially save all of the energy required to compute a 1-D transform.

1.8 Thesis Objectives and Contributions

This thesis attempts to contribute in two ways:

1. Evaluate the potential energy savings of using an MLA-decompression display system.

2. Design and implement such an MLA-decompression system.

1.8.1 Evaluation of Potential Energy Savings

It is important to understand quantitatively the potential savings in energy that might be realized. In this thesis, an attempt to evaluate these energy savings was made by developing a model for the energy cost of the LCD itself as well as evaluating the cost of performing a transform digitally. This is explained in further detail in Chapter 2.

1.8.2 Design and Implementation of an MLA-decompression Display System

This thesis describes the design of a display system that performs a 1-D transform digitally and then uses MLA to implement another 1-D transform to obtain an image on a display. It also outlines the different stages of design that were implemented as well as some of the factors taken into account during the design process.

Chapter 2

Using MLA to Lower Power and Maintain Quality - Practical Considerations

From the previous discussion, it would appear that MLA can provide a useful technique to save energy in systems where a decompression operation is performed. However, in order to evaluate the usefulness of this technique, a couple of factors must be taken into account: energy consumption and image quality.

2.1 Energy Consumption

The energy consumption of the system can be quantitively compared by considering the amount of energy required to compute a single 1-D transform. As explained previously, MLA naturally computes a 1-D transform through the use of the LCD. The other, more regularly used, method to compute this transform would be through the use of digital circuitry or software. Since the LCD takes a single frame period to compute a 1-D transform, the digital circuitry is allowed the same amount of time to compute such a transform. The energy, in μJ, of each different method can then be compared. The following outlines the various methods to compute the transform as well as the nature of the energy consumption of each method. A more quantitative

comparison is given in Chapter 3.

2.1.1 Digital Computation

For an arbitrary transform, one method of performing the transform is to simply use a digital matrix multiplication circuit that is capable of processing the image quickly enough to keep up with the display system frame rate.

Given a wavelet transform, it is possible to implement the transform as a simpler set of operations than the strict matrix multiplication would imply. One method that exploits this and is popular in today's image standards is the Wavelet Lifting Scheme [24].

Matrix Multiplication

The energy cost of a matrix multiplication stems from the multiply-and-accumulate (MAC) operations that occur during computation. To compute a single element of the result matrix, N MAC operations need to be computed. Each of these operations is computed using two b-bit numbers as input and involves a single multiplication and a single addition. In order to keep the same bit-width in the result matrix, the result of the MAC operation is rounded to b bits. In order to compute all N^2 elements of the result matrix, N^3 of these MAC operations need to be computed.

Wavelet Lifting Scheme

The energy cost of the Wavelet Lifting Scheme is similar to the matrix multiplication case as it stems from multiply-and-accumulate operations. However, the wavelet lifting scheme can reduce the overall number of these operations.

2.1.2 LCD Computation

The energy cost of computing a transform using the LCD comes from the charging of the liquid crystal capacitances as well as the driver circuit energy. The cost of charging and discharging the liquid crystal capacitances can be significant given the large area

of the display. The driver circuit energy is heavily dependent on implementation. This energy was not considered initially because of this dependence as well as the assumption that the primary energy cost came from the liquid crystal capacitances themselves.

2.2 Image Quality

Image quality is very difficult to measure quantitatively. One method that is often used is to measure the resultant peak-signal-to-noise (PSNR) ratio of the image. It works as follows.

Given a 'perfect' initial image, A, whatever operations are necessary are performed on A to obtain the result image, B.

$$B = f(A)$$

If the maximum value of A and B is 1, then the PSNR, in dB, of the image is found by computing[5],

$$PSNR = 10 \log \frac{1}{\frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (A_{ij} - B_{ij})^2}$$

where the image is N $\times$ N.

Essentially, part of the PSNR equation is a computation of the average of the squared differences between every element in B and every element in A. This is taken to be a measure of the noise. It then divides the maximum value possible (the peak signal) by this average of squared differences (noise). This ratio is then converted to decibels.

2.3 Maintaining Image Quality

As outlined previously, the passive matrix LCD has a number of requirements placed upon the row and column waveforms so that the resultant image is displayed with-

out distortion. When using MLA to perform decompression as well as display, the resultant image quality must be considered to ensure that it is still satisfactory. To this extent, a couple of techniques to ensure the image quality found in row-at-a-time addressed displays were explored.

2.3.1 Gray shade Generation

One of the components to maintaining image quality is to make sure that gray shade generation is still possible using MLA. For LCDs, even for ones that use MLA, there are a number of techniques to generate gray shades.

Frame Modulation

A simple technique is known as frame modulation [13]. This involves creating a super-frame out of a number of sub-frames. Leaving a pixel black for some fraction of the total number of frames and white for the rest will, on average, yield a gray pixel. While this technique is relatively simple to implement, it can lead to excessively high refresh rates to maintain an acceptably fast overall image refresh rate. In other words, the sub-frames must be fast enough such that combining multiple sub-frames into a single frame allows the LCD to respond to the single overall frame and not the individual sub-frames. In general, this technique is useful for yielding up to about 16 gray levels [14].

Full-Amplitude Pulse Height Modulation

Another technique is known as full-amplitude pulse height modulation [15]. It works as follows:

Previously, during the discussion of a MLA-driven LCD, it was assumed that all pixel values were +/- 1 and the resultant pixel value was only dependent on its associated image value. However, if the image values are not all +/- 1, the last term in the expression cannot be simplified to a constant as it was before in equation 1.1.

$$P_{ij_{rms}} = \sqrt{\frac{1}{N}(NF^2 - 2cNF^2A_{ij} + \sum_{t=0}^{N-1} G_j(t)^2)} \tag{2.1}$$

Clearly, the pixel now depends on all of the image values in its column. To correct for this, an extra row period can be added. During this row period, all rows will take on the value zero while each column will assume the value of:

$$\sqrt{N - \sum_{t=0}^{N-1} G_j(t)^2} \tag{2.2}$$

In this way, this value will end up being squared and added under the square root from above. This term will then cancel the $\sum_{t=0}^{N-1} G_j(t)^2$ term leaving only the constant N term and eliminating the dependence on other column values.

2.3.2 Correction for Biorthogonal Wavelets

When biorthogonal wavelets are used, the resultant row matrix has non-orthogonal rows[22]. The non-orthogonality of the row matrix means that

$$\sum_{t=0}^{N-1} F_i(t)F_j(t) \neq 0, i \neq j$$

This, in turn, implies that the RMS pixel voltage will depend not just on A_{ij} but other image values as well, causing distortion and loss of quality. In order to correct for this, a simple approach is proposed.

Since the resultant row matrix is non-orthogonal, $F^T \neq F^{-1}$. Therefore, in order to obtain the image, A, the transformed image, F^TAF, needs to be post-multiplied by F^{-1}[11] and pre-multiplied by $(F^T)^{-1}$. The post-multiplication happens digitally while the pre-multipliation occurs at the LCD. This means that the row driving matrix, normally just F^T, must instead be the matrix $(F^T)^{-1}$.

Chapter 3

Matlab and Printed Circuit Board Prototypes

The display driver design was completed in stages, moving from software prototype to PCB prototype and finally to implementation as an integrated circuit. The Matlab and PCB prototypes are detailed below.

3.1 Matlab/C Implementation

In the Matlab implementation, the transform using the LCD was simulated by building a model for the display in Matlab and iterating the model through a single frame period. The digital transform using the lifting wavelet scheme was implemented in C by using the transform component of Jasper [1], a software implementation of the JPEG2000 standard.

3.1.1 Matlab Display Simulation

The display was modeled as an array of fixed capacitances whose values were estimated based on the characteristics of an actual display: a transflextive graphic display module with 64 rows and 128 columns, the Microtips MTG-F12864AFYHSAY-10A. For ease of implementation (ie. only dealing with square matrices), the display was

driven as a 64x64 display. The pixel capacitance was estimated as follows:

$$C = \frac{A\epsilon_r\epsilon_0}{t} \tag{3.1}$$

Since the ϵ_r of the liquid crystal changes with applied electric field, a mean of 7 was used to simplify calculations. The mean of t, the thickness, was estimated at $5\mu m$, a typical distance between the row and column lines. Using these values, individual pixel capacitances were calculated. Later on, the total capacitance of all pixels was measured and the individual pixel capacitance inferred.

The display model was iterated through a single frame period to determine both the energy consumed as well as the resultant value of each of the pixels. Using these pixel values as well as the original image, the PSNR was computed as a measure of the quality. The energy was computed assuming each pixel capacitance charged and discharged fully during each row period. For example, if a pixel were at 3V during one row period and 4V during the next, the total energy consumed would be computed as $(3^2 + 4^2)C$. This energy computation was chosen as a worst-case value since driver architectures could be chosen to obtain smaller pixel energies than this but none could consume more than this.

Matlab/C Results

The energy consumed by the display depends on the image data since the pixels rest naturally in either the black or white state, depending on the type of liquid crystal, but require energy to keep them flipped in the opposite state. Because of this dependence on image data a fixed test image as shown in Figure 3-1 was chosen for energy and quality measurements. Various row matrices, based on a given wavelet, and at different scales of resolution, were tested for both energy consumed (per frame) and resultant image quality. The results are summarized below.

The wavelets with large PSNR (>200) are all those that are orthogonal. Recalling that each pixel will display the correct value given an orthogonal row matrix, these results appear consistent. The biorthogonal wavelets (BIOR) use $(F^T)^{-1}$ as the

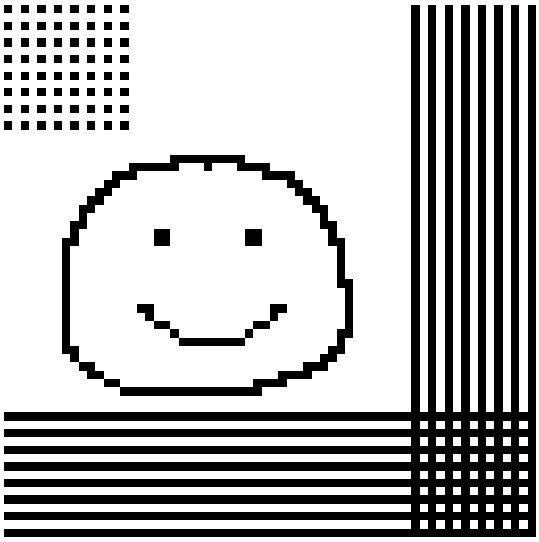

Figure 3-1: Test Image

row driving matrix which greatly improves the quality. Unfortunately, driving with $(F^T)^{-1}$ does not remove all of the sources of distortion in the resultant image.

The energy consumption of different wavelets is approximately the same, which is also a consistent result. If a pixel capacitance is charged and discharged fully each row period, then the RMS voltage can be used as a measure of the energy consumed. This RMS voltage also represents the pixel value which should be the same regardless of wavelet since the resultant image should be the same regardless of the wavelet used. Another point to note is that the energy required for multiple-scale decomposition and single-scale decomposition is the same. The reason for this is that, as explained previously, a multiple-scale decomposition can be formed with a single matrix since it is the product of a series of single-scale matrices.

3.2 SA1100/Jouletrack

An excerpt from Jasper, an implementation of the JPEG2000 standard was used. This program was run using JouleTrack [20], an SA1100 microprocessor simulator, which reported the amount of energy consumed in the computation. Unfortunately, it was possible to perform a transform using only the BIOR-2.2 wavelet. However, in

Table 3.1: Matlab Simulation Results

Wavelet	Scale	Energy (μJ)	PSNR (dB)
DB1	1	2.77	291
DB1	2	2.77	293
DB1	3	2.77	294
DB2	1	2.77	249
DB2	2	2.77	242
DB2	3	2.77	238
BIOR-2.2	1	2.92	18.9
BIOR-2.2	2	2.97	14.9
BIOR-4.4	1	2.79	29.4
BIOR-4.4	2	2.80	22.8

the lifting transform, this wavelet results in multiplications by powers of 2 (a simple operation to compute), which should therefore provide a lower-bound on the energy consumed by this method.

The energy consumed by a single 1-D transform on the test image was 318.6μJ.

From these simple implementations, it became clear that great savings in energy without great loss in display quality could be obtained.

3.3 Printed Circuit Board (PCB) Prototype

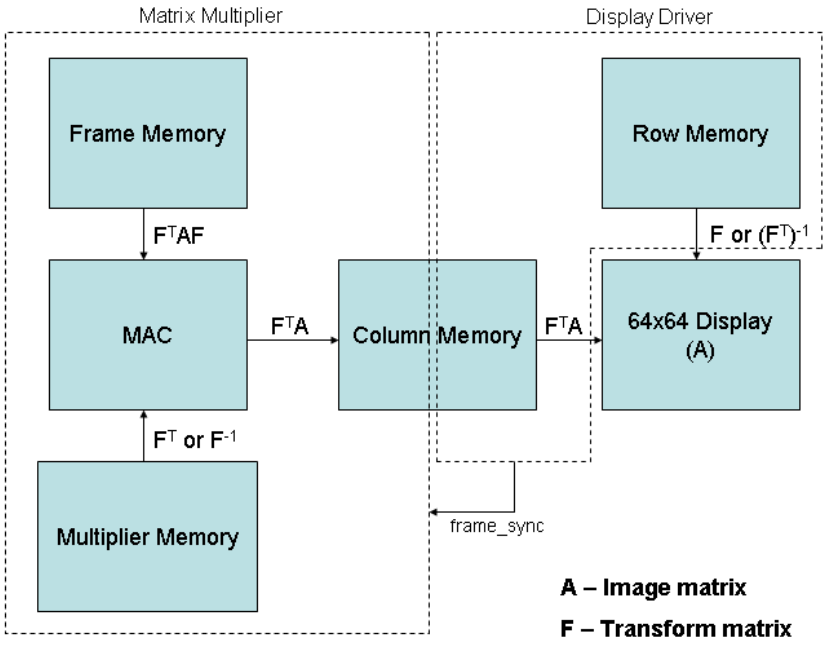

Figure 3-2: PCB Architecture

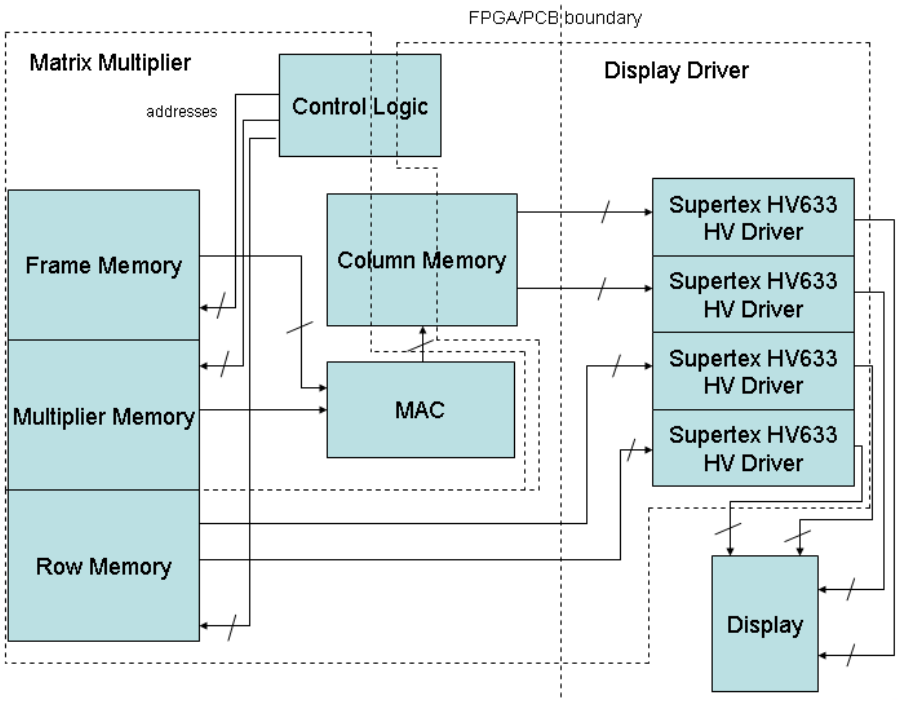

Figure 3-3: PCB Block Diagram

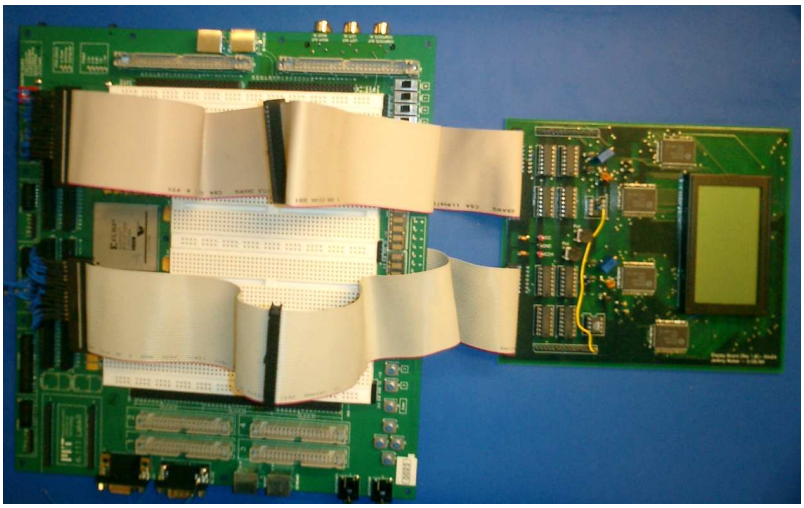

Figure 3-4: PCB System Photo

In order to prototype the proposed idea before building an integrated circuit, a PCB was constructed. The digital 1-D transform was implemented as a matrix multiplication in Verilog on a Xilinx Field Programmable Gate Array (FPGA). The FPGA also controlled a set of 7-bit high-voltage drivers that in turn drove a passive matrix LCD to perform the other 1-D transform. The system as a whole was intended to perform the inverse transform step of the decompression process, the result of which was shown on the LCD. The system was designed to run in two different configurations: display only and transform and display.

The digital transform was implemented as a generic matrix multiplication so that any arbitrary transform at any scale could be computed. This matrix multiplication was accomplished by using a single multiply-and-accumulate (MAC) unit running fast enough to compute N^3 multiplications and additions in the required time. Four separate random access memories (RAMs) were required: one to hold the row matrix that drove the rows of the display, one to hold the column matrix that drove the columns of the display, one to hold the transformed image matrix that served as input, and one to hold the matrix that was multiplied with the transformed image to perform a 1-D transform. An interface to the display driver chips was also implemented. This interface read data from the row and column RAMs and sent it out to the display driver chips asynchronous to the matrix multiplication. An asynchronous design was chosen for modularity and to make testing simpler. The column RAM acted as the asynchronous interface between the digital matrix multiplier and the display driver interface. It was implemented as an asynchronous dual-port RAM.

At the highest level, the system can be broken into two subsections: display driver and matrix multiplier.

3.3.1 Display Driver

The display driver section consists of digital control and interface logic for the display driver chips, supporting analog circuitry for the display driver chips, and the LCD itself.

Physically, the digital control runs from a Xilinx Virtex-II FPGA and connects

to a secondary board with level shifters, a high-voltage ramp generator, the display driver ICs (Supertex HV633), and the 64x128 display. Each display driver IC has 32 high-voltage outputs capable of delivering voltages as high as 80V with a precision of 7 bits. However, while the chips can support voltages of up to 80V, only a range of up to 30V was required for this application. To drive the display, 4 ICs are used in total: 2 to drive 64 total rows, and 2 to drive 64 total columns. The unused columns are left disconnected.

3.3.2 Matrix Multiplier

The matrix multiplier section consists of user input logic, a single multiply-and-accumulate (MAC) unit to perform the matrix multiplication, control logic, and memories to store the row and column waveforms as well as the matrices being multiplied.

The input logic consists of 3 switches: *reset*, *display*, and *load-control*. *reset* is a general asynchronous reset for the whole system. *display* controls whether to simply display the pre-loaded values in the row and column memories. *load-control* simply loads in a set of control bits that determines whether there are any row-periods that need to be addressed during the frame-period.

A single MAC unit is used to compute an entire matrix multiplication. This decision saved on area and ensured simplicity in implementation. Although this configuration means that a single MAC unit must compute N^3 multiplications, this number of multiplications needs to be computed only in time for a single frame period, which is relatively long; a 100 Hz frame refresh rate gives a frame period of 10ms. For a 64×64 display, each multiplication/addition can take 38ns, yielding a clock frequency of only 25MHz.

There are 4 memories in total: row, column, frame and multiplier. The frame memory is designed to hold a number of different input images for use possibly in a very short video. Since it is implemented using the FPGA, its size can be arbitrary up to the limits of the FPGA. The multiplier memory is used to store the matrix that will be post-multiplied with the current matrix accessed in the frame memory. The row memory stores the waveforms that will be applied to the rows of the LCD. Similarly,

the column memory stores the waveforms that will be applied to the columns of the LCD. This column memory is write-accesible to the MAC unit and read-accessible to the display driver logic.

3.3.3 Results

Below is a number of photos of the display using various wavelets to perform the image transform.

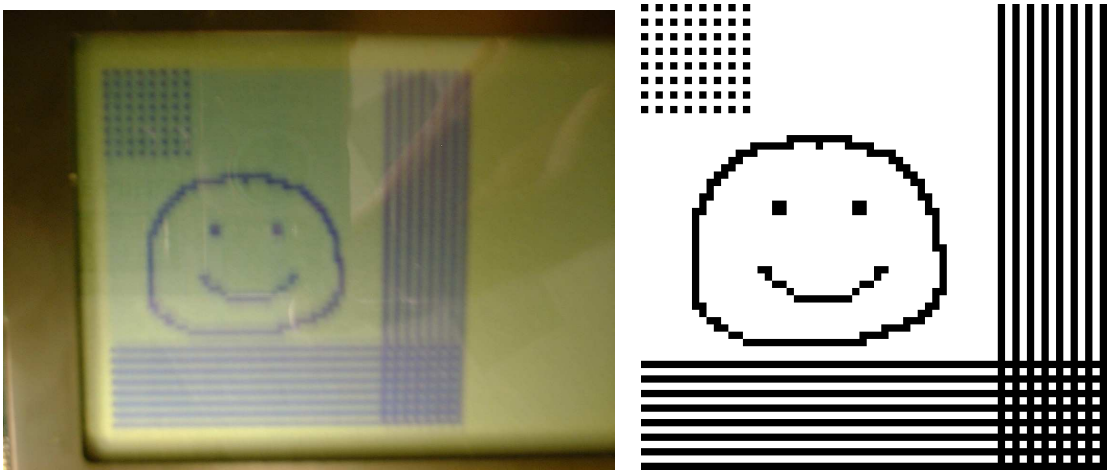

Figure 3-5: PCB vs. Matlab : Using DB1 Wavelet (1 scale of decomposition)

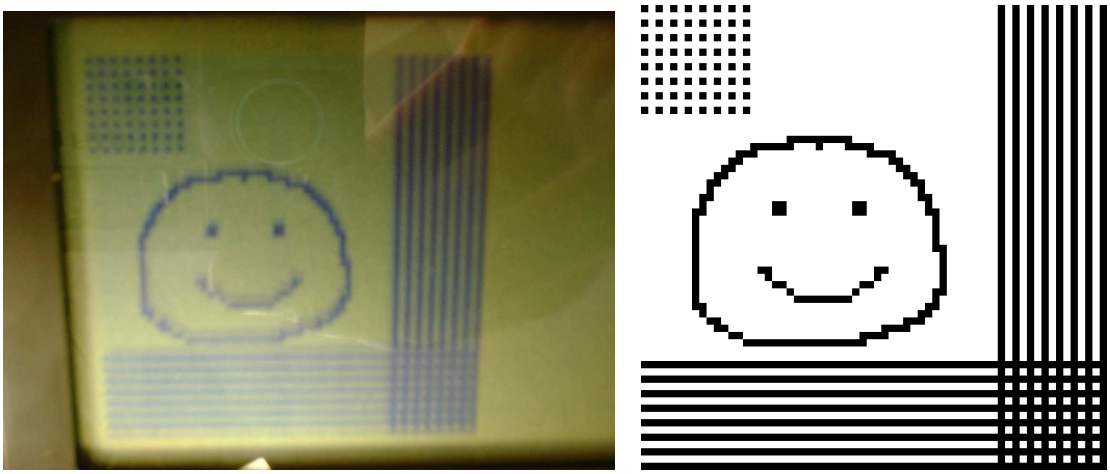

Figure 3-6: PCB vs. Matlab : Using DB1 Wavelet (3 scales of decomposition)

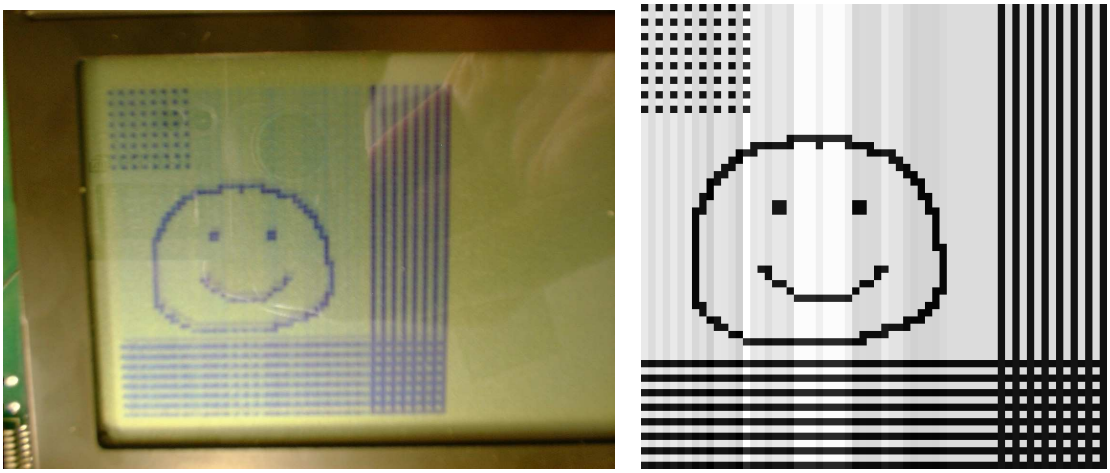

Figure 3-7: PCB vs. Matlab : Using Bior2.2 Wavelet (1 scale of decomposition, driving with $(F^T)^{-1}$)

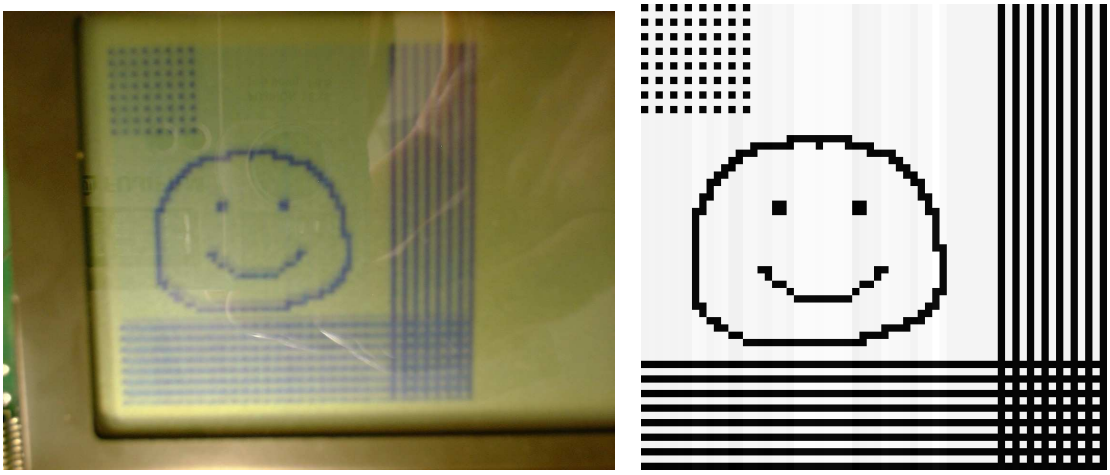

Figure 3-8: PCB vs. Matlab : Using Bior4.4 Wavelet (1 scale of decomposition, driving with $(F^T)^{-1}$)

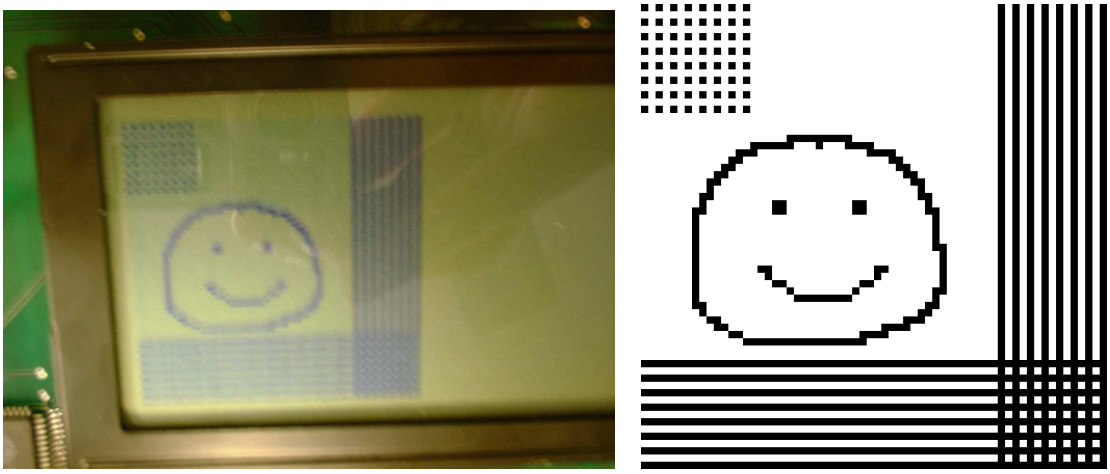

Figure 3-9: PCB vs. Matlab : Using DB2 Wavelet (2 scales of decomposition)

Unfortunately, given the time, it was not possible to set up a system to measure the PSNR of the resultant images on the LCD so only qualitative comparisons can be made. However, from the images, it appears that the Matlab implementation closely models that of the actual LCD. In particular, the distortion in Figure 3-7 in the Matlab output is also clearly seen in the PCB implementation. It also appears that while the orthogonal wavelets have exceptional quality and hence, very high PSNR, it is difficult for the eye to discern the errors that result from having a PSNR of almost 30dB as is the case for the Bior4.4 Wavelet shown in Figure 3-8.

Chapter 4

Integrated Circuit Design

The IC (chip) was designed to resemble the PCB system very closely; this made the transition from one to the other a relatively simple process.

4.1 Architecture

The integrated circuit can be broken into two pieces: a matrix multiplier and a display driver. These two pieces communicated with each other via an asynchronous off-chip RAM (column memory) that acted as output for the matrix multiplier and as input for the column drivers of the display driver. The *frame-sync* signal allowed the matrix multiplier to know when the display driver was starting the display of a new frame and therefore allowed somewhat synchronous operation of the two pieces.

The matrix multiplier took in the transformed image, normally $F^T AF$, and a multiplication matrix, either F^T or F^{-1}, and computed the matrix post-multiplication to give $F^T A$. This output of the matrix multplier was used as the column waveform data for the display driver.

The display driver took in the row waveform matrix, either F or $(F^T)^{-1}$, the column waveform matrix from the output of the matrix multiplier, and drove the display over a frame period with these two matrices. The result was displayed on the LCD.

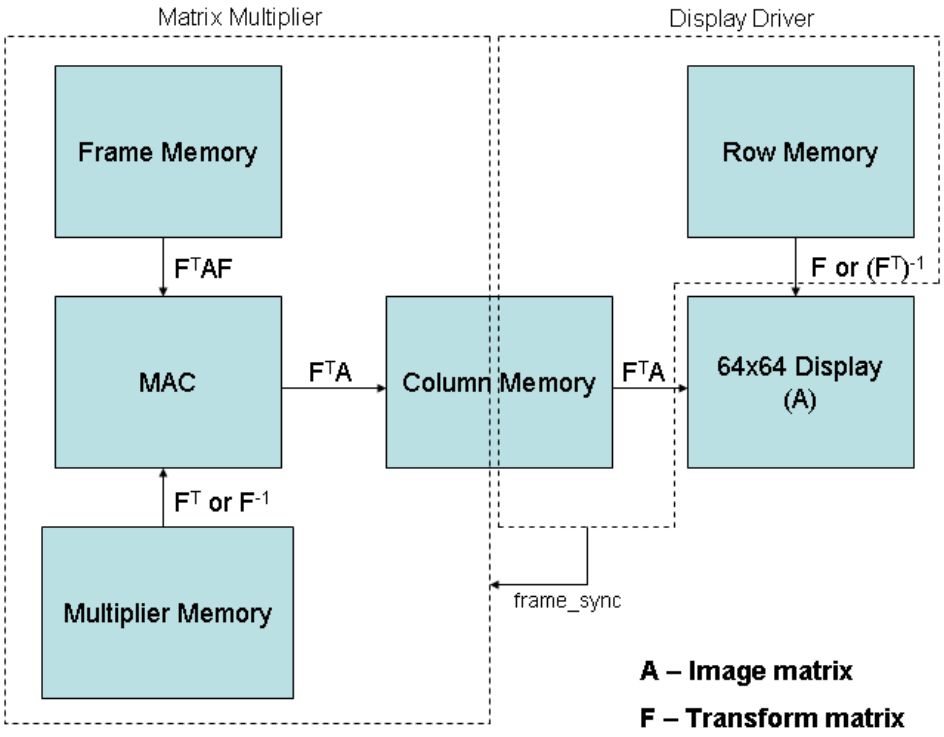

Figure 4-1: Integrated Circuit Architecture

4.2 Detailed Design

Figure 4-2 shows the various blocks that composed the IC as well as the defining boundaries of the two larger components that make up the design: the matrix multiplier and display driver.

The matrix multiplier section consisted of a number of off-chip RAMs to store the frame data (frame memory), the matrix to be multiplied with the frame data (multiplier memory), and the column data matrix (column memory). In addition to this was a MAC unit to perform the operations necessary for a matrix multiplication, and a control logic unit that generated the addresses for the off-chip RAMs.

The display driver section consisted of a number of off-chip RAMs that stored the row (row memory) and column (column memory) waveforms to be used during the current frame. On-chip it consisted of a block of control logic that generated the addresses for the RAMs and other control signals, as well as a series of driver circuits

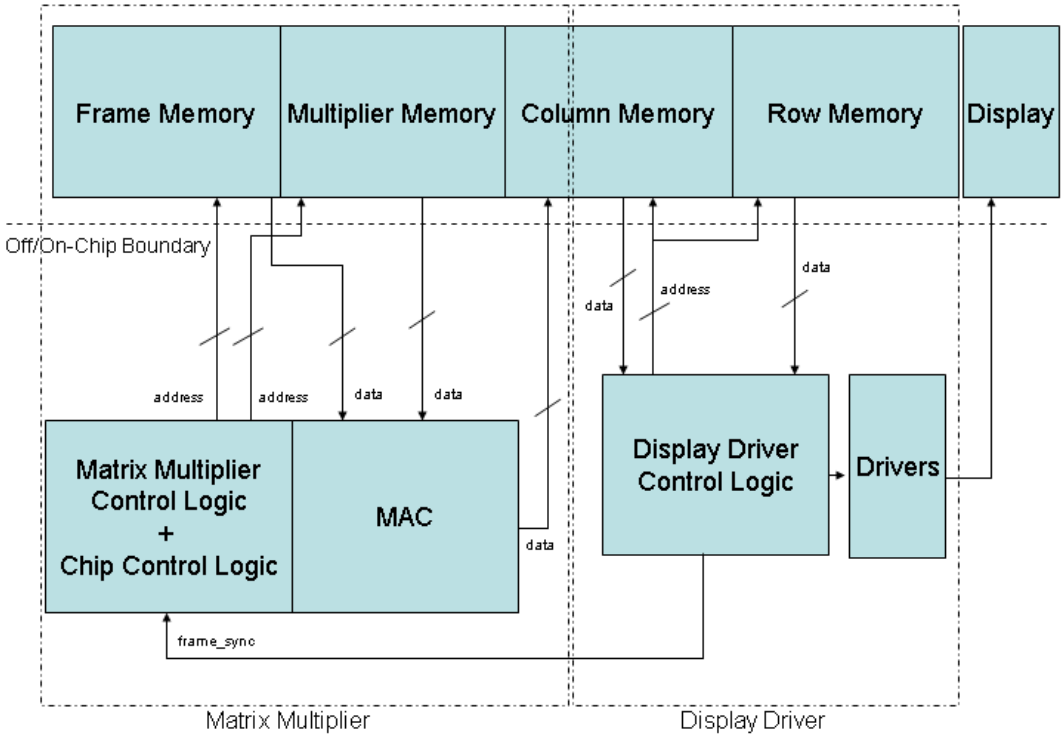

Figure 4-2: Integrated Circuit Block Diagram

that handled digital-to-analog (D/A) conversion and analog buffering.

4.2.1 Chip Control Logic

The chip control logic consisted simply of a set of registers that held control bits used in various parts of the chip as well as two reset signals. The reset signals indicated whether the entire chip was in reset mode, loading-control-bits mode, or run mode.

There were 17 control bits total although only 16 of them were used; one bit was used to determine when all of the control bits had been loaded. The other bits were used to determine if there were any extra row periods to take into account beyond the 64 physical rows of the display. This extra row period would be used for gray-scale images. Other bits were used to determine how often to run the matrix multiplier. If a static image were shown on the display, the matrix multiplier wouldn't need to recompute the same multiplication over and over in order to display the image.

The reset signals were *reset*, which reset the entire chip into a known state, and *control-reset*, which reset everything but the logic that allowed the control bits to be loaded.

4.2.2 MAC Unit

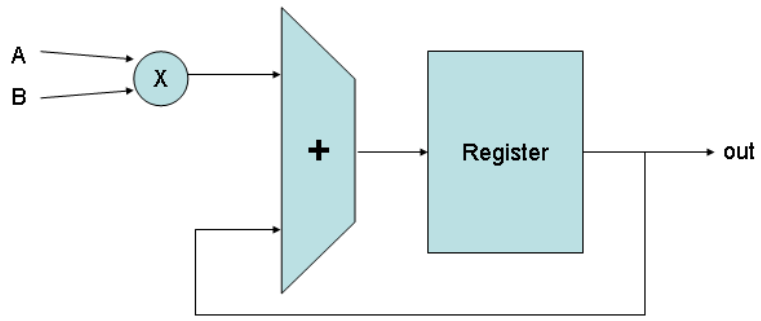

Figure 4-3: MAC

The matrix multiplier was built as a serial one in that it consisted of a single MAC unit that ran quickly to perform a matrix multiplication. A number of possible alternatives could have been chosen such as a 64-element-wide MAC unit or a fixed number of MAC units (just enough to perform the required number of MAC operations for a given transform). However, the serial matrix multiplier was chosen because of its small size and flexibility in performing any possible matrix multiplication. The bit-width of 8 bits was chosen since the analog output driver specification was only up to 8 bits.

The MAC unit consisted of a single adder (16 bit) and a multiplier (8 bit by 8 bit) as well as an extra-wide (16 bit) accumulate register to store partial results.

While the accumulate register was 16 bits wide in order to accomodate the output of the multiplier and adder, only the upper 8 bits were actually output to the column memory since this was the bit-width of the analog driver circuitry. Both adder and multiplier blocks were synthesized from Verilog code to meet timing specifications.

4.2.3 Matrix Multiplier Control Logic

The control logic for the matrix multiplier was a simple state machine on which a number of different RAM addresses were based.

The state machine had the following states: clear0, load display data, clear1, load grays, clear2, run matrix multiplier, clear3, run just display, clear4. The clear states were necessary to allow counters and such to reset to their correct values. The rest of the control logic was logic used to generate addreses for the off-chip RAMs.

4.2.4 Display Driver Control Logic

The control logic for the display driver consisted of a simple state machine, an asynchronous reset, an address counter for the off-chip RAMs, a high-speed D/A converter counter, as well as a 2's complement to unsigned converter that was used to implement frame inversion.

The state machine kept track of the phases of operation: *reset*, *convert*, *settle*, and *shift*. *reset* was a general reset state. During *convert*, the D/A conversion was taking place. *settle* was used to indicate the period of time when the driver outputs were settling to their final values. During this settling state, new driver data was also being shifted in for the next frame. Finally, *shift* was the state when only new driver data was being shifted in for the next frame. During this state, all driver outputs were to have settled.

The D/A converter counter ran at 42MHz, which enabled counting from 0 to 255, and hence conversion, in about 6μs or 2.9

The 2's complement to unsigned converter was necessary so that the signed numbers could be converted to values from 0->255 that the D/A converter would under-

stand. This converter also took as input an invert signal that told it whether the data needed to be inverted first. This inversion enabled the driver to put a net 0V DC across the pixels, a requirement necessary not to destroy liquid crystals[16].

4.2.5 Drivers

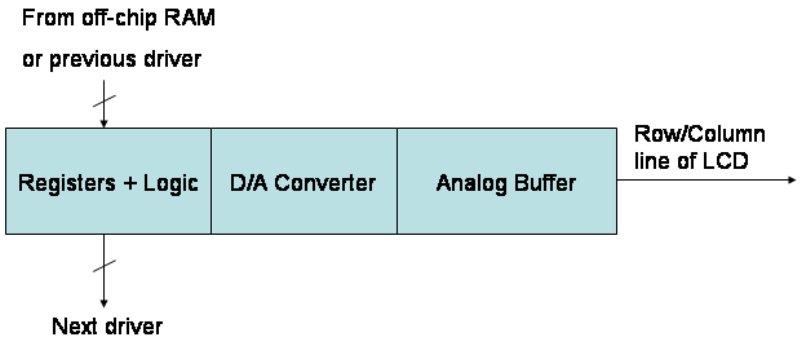

Figure 4-4: Single row/column driver block diagram

Each driver consisted of an 8-bit register to store the digital row/column value for the current row period, a D/A converter to convert that value to an analog voltage, an analog buffer to buffer that voltage and drive it on the row/column line of the display, as well as a few control registers and logic associated with control of the previous parts.

Registers + Logic and D/A Converter

The D/A converter consisted of an 8-bit register as well as control logic that turned on and off a set of pass transistors. Its general operation was designed similarly

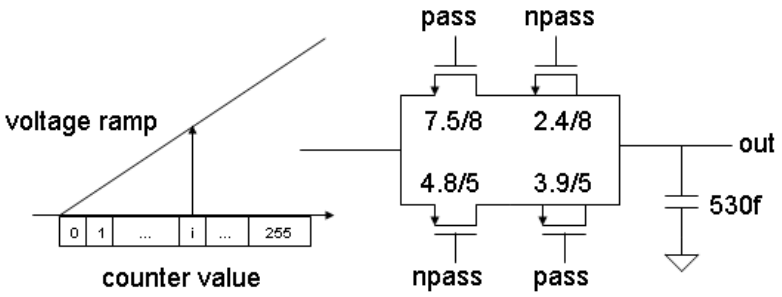

Figure 4-5: D/A Converter

to the high-voltage drivers used in the PCB implementation[6]. The pass transistors (Figure 4-5) were used to pass a high-voltage ramp signal until the counter value, from the control logic, equaled the digital value to be output. Once the pass transistors were turned off, a capacitor held the value. The 8-bit register that stored the digital data was configured as a shift register as part of a chain comprising all the other output drivers on the chip. This allowed for a simple serial mechanism to bring the driver data into the chip.

The digital section that controlled whether the pass transistors were turned on consisted of a digital comparator that compared the digital input to the current counter value as well as a register to synchronize this comparison. This register was necessary since the counter value would arrive at each individual row/column driver at a different time because of wire delays. By synchronizing this value with this register, it could be ensured that each driver would turn off its pass transistors at the

same time.

The pass transistor circuit consisted of the usual NMOS and PMOS devices to allow full output voltage range with a few additional transistors: an additional NMOS and an additional PMOS device were added in series with the signal path to help cancel charge injection [21]. These dummy transistors had both drain and source connected to the signal line, creating a short-circuit. Their use, however, was to absorb some of the charge injection that occurred when the NMOS and PMOS pass transistors were turned off. When the pass transistors were turned off, charge would flow out from under their channels into the source and drain regions[19]. Charge would also flow through the gate-source/gate-drain capacitances of the pass transistors[19]. Since the sources/drains of the pass transistors were connected to the output node, this could cause a voltage shift from the correct value. The dummy devices were configured such that they turned on when the pass transistors were turning off. In this way, the dummy devices attempted to absorb the extra charge injection into the output node.

There were a number of design factors that contributed to the various sizes of the pass transistors and dummy devices. In general, the dummy devices should have been half the width/length of the pass transistors since each pass transistor would normally inject half of its channel charge into the output node[23]. However, the pass transistors were more difficult to size. Since the transistors were all high-voltage devices, they had relatively large (on the order of 1kΩ) series resistances in their sources and drains. These resistances in combination with the output capacitor created a series resistor-capacitor circuit (capacitor to ground) between the input ramp voltage and the output node. The static error of this circuit to an input voltage ramp is given by RC. Thus, to reduce the error, a reduction in the resistance or the capacitance was necessary. This step makes sense intuitively since the time constant of the system becomes smaller, allowing the output to follow the input signal more closely. However, a smaller capacitor poses a problem since a large one is necessary to prevent leakage effects from being significant and also to absorb any residual charge injection. A smaller capacitor is also more susceptible to capacitive coupling from other nodes

in the IC. Reducing the resistance also poses a problem since this reduction would require that the width of the devices be increased. However, this increased width also increases the size of the channel as well as the gate-to-source capacitance. Each of these in turn increases the charge injection. An optimum for all devices was found through iteration.

Analog Buffer

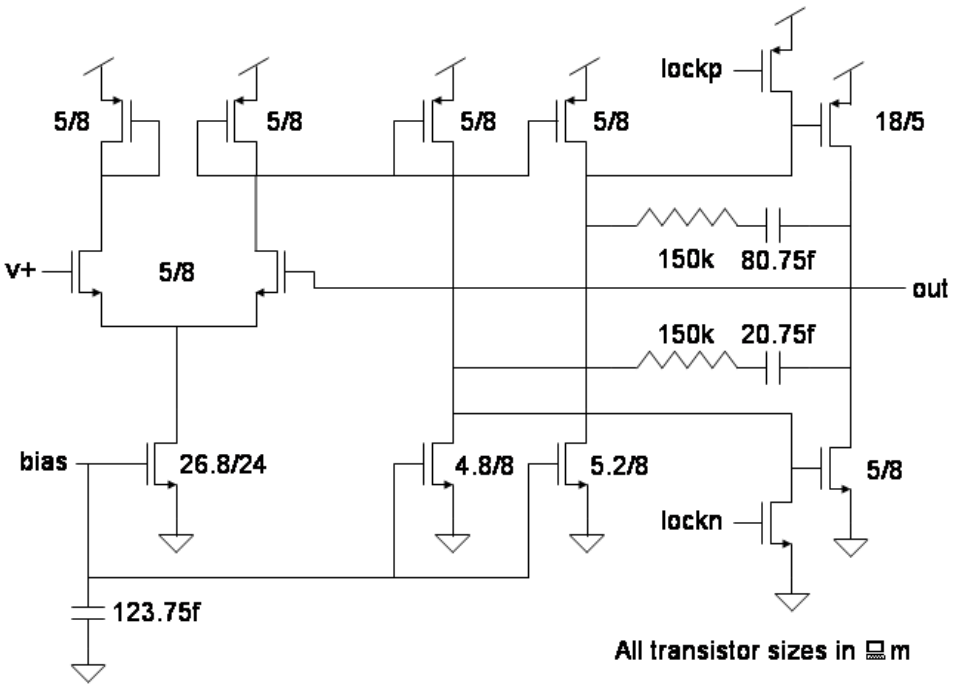

Figure 4-6: Class B Buffer

The analog buffer was essentially a class-B opamp configured for unity gain [27]. It consisted of a single differential input stage, two common-source gain stages and a class-B output stage.

Class-B operation was desired since large currents were necessary to slew the LCD capacitances but only for a short period of time. Class-B insured no power-hungry class-A bias currents at the output stage. In addition to a class-B output stage, the

class-A bias currents that were used in the rest of the circuit could be shut off once all of the drive voltages had settled. Therefore, once the D/A conversion was complete and all row/column voltages were at their final values, the bias currents could be shut off to save power.

Class-B operation of this circuit was ensured by sizing the NMOS devices of the common-source gain stages so that the output voltages turned the output stage NMOS and PMOS devices off when both the input and the output were equal. The major source of DC error in this circuit was a result of the single transistor current source in the differential stage. This caused a difference in the output and input voltages when the input was at relatively low voltages (ie. < 5V). At such low voltages, this NMOS transistor was not fully saturated and therefore had a finite output impedance which, in turn, caused a mismatch between the output and input voltages.

Dynamically, this circuit was very difficult to use given that the output node could see anywhere from zero capacitance to a very large capacitance from the LCD. This difficulty stemmed from the fact that the outputs were tied to only one end of a pixel capacitor. Not only that, but each common-source stage added a single pole of its own in addition to the pole at the output stage. These poles from the common-source stages were close to one another since the output impedances at these nodes were similar. In order to properly compensate this system, it was necessary to introduce a significant constant output capacitance that could act as a dominant pole. In addition to this, minor loop compensation capacitors and resistors were added from the output back to each common-source stage. Each capacitor-resistor pair added a left-half-plane zero which improved the phase around the location of each pole from the common-source stages.

4.2.6 Area Constraints

With no area constraints, the IC would have had a digital matrix multiplier and a number of high-voltage analog outputs to drive the rows and columns of the display. Unfortunately, area constraints allowed only 32 output drivers per IC. Because of this, 4 ICs in total were required to drive the entire display, while only one matrix

multiplier from a single IC was used. In order to cut down on the number of off-chip RAMs that 4 ICs might require, the ICs were designed to share the same row and column data bus. To facilitate this sharing, some extra logic was added to the design.

4.3 Results

A number of simulations were run on the post-layout and extracted circuits. The most pertinent results are in the following sections. Further simulations of the analog circuits can be found in Appendix A.

In each table below, the total energy refers to the energy for one frame period and all of the elements in the system of that type. For example, in Table 4.2, Row/Column Drivers refers to all 128 drivers that would be required to drive the display, even though these 128 drivers would be spread across 4 ICs.

4.3.1 Matrix Multiplier Energy Consumption

The energy consumption for a single matrix multiplication was simulated in Nanosim and measured to be 118.7μJ. This includes the energy taken up by all on-chip circuitry but does not include the energy consumed by off-chip RAMs or that consumed by output buffers required to drive signals off-chip.

4.3.2 Display Driver: Analog Energy Consumption

Table 4.1: Display Driver: Analog Energy Consumption

Element	Energy/Driver (μJ)	Total Energy
D/A + Analog Buffer	50nJ/row period	409.6μJ

The above table was generated using a 210μs row period, 105μs conversion time (ie. the time before all bias currents were shut off), and a 6μs D/A conversion time. This measurement does not include the energy that would be consumed by the ramp voltage required for the D/A converters.

The energy consumption of the analog components of the display driver are largely taken up by the analog buffer. This energy consumption is in turn set by 2 currents: bias and transient. The bias current can actually be adjusted since the ability to lock the output voltage at a specified time was made part of the circuit. In this way, the bias current can be shut off once the output voltage has settled.

4.3.3 Display Driver: Digital Energy Consumption

Table 4.2: Display Driver: Digital Energy Consumption

Element	Energy Consumption
Control Circuitry	38.4μJ
Row/Column Drivers	144μJ
Total	182.4μJ

This table does not include the energy that would be required by the off-chip RAMs or output buffers required to drive signals off-chip.

From Table 4.2, it seems clear that the energy of the digital circuitry for the row/column drivers dominated that of the control circuitry. For this reason, the energy consumption of the digital circuitry for the row/column drivers was investigated further. In the analysis below, various elements that compose the row/column driver circuitry were examined. Unfortunately, due to the setup of the design, only the entire row/column driver could be simulated at one time. Back-calculations involving typical energy characteristics for each component were then used to determine the major sources of energy consumption.

Table 4.3: Row/Column Driver Energy Consumption

Element	Characteristics	Current/Driver	Total Energy
DFS8 (D flip-flop)	381.2fA 1->1 transition 42MHz clock	16.01μA	91.12μJ
JKSA2 (JK flip-flop)	8μW/MHz 303kHz	5.82μA	33.12μJ
AN211 (AND/NOR)	3.49μW/MHz 1.5 elements worth 42MHz, 2.9% of time	1.93μA	10.98μJ
IN2 (Inverter)	2.54μW/MHz 127/256 elements worth 42MHz, 2.9% of time	465nA	2.65μJ
ON22 (OR/NAND)	1.81μW/MHz 3/4 elements worth 42MHz, 2.9% of time	501nA	2.85μJ
Total	computed		140.76μJ
Total	nanosim		144.5μJ

From this analysis, it appears that the non-transition switching energy from the D flip-flop is a significant drain on the overall energy. If this clock signal were gated the energy consumption could be reduced a great deal since this flip-flop is required only during the D/A conversion phase of the row period, a small fraction of the overall row period time.

Chapter 5

Conclusions and Future Work

The purpose of this thesis was to create a low power display driver that still maintained high quality images. The technique of using MLA as a way of obtaining a free image transform was explored since this allowed the elimination of this operation in the digital domain. The system constructed consisted of a digital matrix multiplier and a set of drivers to drive the rows and columns of the display. The digital matrix multiplier, used to perform a single 1-D transform, consisted of a single MAC unit operating quickly enough to compute the whole matrix multiplication in one frame period. Each row/column driver consisted of a set of shift registers to store the digital data, a D/A converter and an analog buffer. A ramp + counter-based D/A converter was used to eliminate static power consumption in the conversion part of each driver. A class-B output buffer was used to again eliminate static power caused by a class-A output stage in each driver. In addition to the class-B output stage, transistors were added to shut off the bias currents in the rest of the buffer circuit.

5.1 Conclusions

5.1.1 Energy Consumption

The following table summarizes the energy consumption of each major section of the design and indicates the components in each section that consume the majority of

that energy.

Table 5.1: Energy Consumption of the IC

Element	Energy Consumption (one frame)
Digital Matrix Multiplier	118.7μJ
Driver Control Circuitry	38μJ
Drivers (digital)	144μJ
Drivers (analog)	409.6μJ
Display Capacitance	2.77μJ
Total (Display driver circuitry)	592μJ

The energy consumption of the matrix multiplication system was 118.7μJ compared with a total cost of driving the display of 592μJ.

From the table above, it is clear that the energy consumption of the display driver system was larger than that of the matrix multiplier. This suggests that the usage of the display to perform an image transform may not be the most efficient area in which to develop the design. However, it should also be noted that this matrix multiplication is inherent to the display driver system itself, and essentially comes for free. Thus, the total savings can be computed as the energy of the 1-D transformation that is saved, divided by the total energy of a system that does not use the LCD to perform a 1-D transform.

$$\frac{118.7}{592 + 2 * 118.7 + 2.77} = 14.3\% \tag{5.1}$$

These savings, while not insignificant, are nowhere near those originally predicted by the Jouletrack and Matlab simulations. The reason for this stems from the assumption that the primary energy consumption would come from the display capacitance itself. In reality, the display driver energy was the most significant.

5.1.2 Quality

Another objective of this thesis was to maintain the quality of the displayed images. A number of techniques were required in order to mitigate the effects of non-orthogonal

driving matrices as well as to create gray levels.

From the PCB prototype, it is clear that the quality is dependent largely on the orthogonality of the driving matrix. Other distortions undoubtedly arose from the ability of the high-voltage drivers to produce accurate voltage levels as well as producing them quickly enough so that the RMS of the voltage was accurate.

5.2 Lessons Learned

5.2.1 High-Voltage Devices

The high-voltage devices used in this thesis greatly limit the performance of the output buffer circuit. Because they are able to withstand high voltages, their sizes are generally quite large and their g_m/I ratio tends to be fairly low due to a larger gate oxide thickness. Because of their large size, large parasitic diodes exist that add capacitance at all nodes in the circuit. In general, all of these characteristics cause these devices to be big and slow; precisely the opposite of what is desired in a driver circuit.

5.3 Improvements and Future Work

5.3.1 Process

The current 0.8μm process that was used in this project is not the latest available from Austriamicrosystems. A smaller linewidth process that also has a high-voltage option is available. This smaller process allows smaller high-voltage transistors which would in turn result in smaller parasitics and hopefully improved performance.

5.3.2 Transform Used

The transform that is used in the system determines two things: the resultant compression of the image, and the resultant driving voltages from the row and column drivers. If a measure of the resultant compression a given transform might yield is

used, it would be possible to select a transform that maximizes compression, minimizes driving voltage, as well as minimizing distortion caused by non-orthogonality. By minimizing the driving voltage, it would be possible to lower the energy consumed by the driver circuits as well as possibly to move to a lower-voltage process resulting in smaller, faster transistors.

5.3.3 Interframe Compression

Another way to lower energy consumption is to lower the bandwidth of the system, ie. lower the amount of data that needs to be sent to the display system at any given time. This could be possible through the use of interframe compression. This idea has already been explored in the context of an OLED display [8].

5.3.4 Organic Light-Emitting Diode (OLED) Displays

OLEDs are hailed as being the up-and-coming display technology of choice. This thesis, on the other hand, explored only the use of passive matrix LCDs as the display technology. The results obtained are therefore not readily applicable to OLEDs. However, while this thesis explored the use of the RMS response of the liquid crystal to perform computation, with an OLED it might be possible to use their fast response time and the eye's ability to take an average to do the same. Using an OLED with at-pixel multiplier circuitry as well as the eye's natural ability to perform averaging, a matrix multiplication could be obtained just the same as with an LCD.

However, one question left unanswered in this case is that of negative numbers. The matrix multiplication uses negative numbers and therefore subtraction. However, the eye cannot perform subtraction and so another means must be found to work around this problem.

Appendix A

D/A + Analog Buffer Simulations

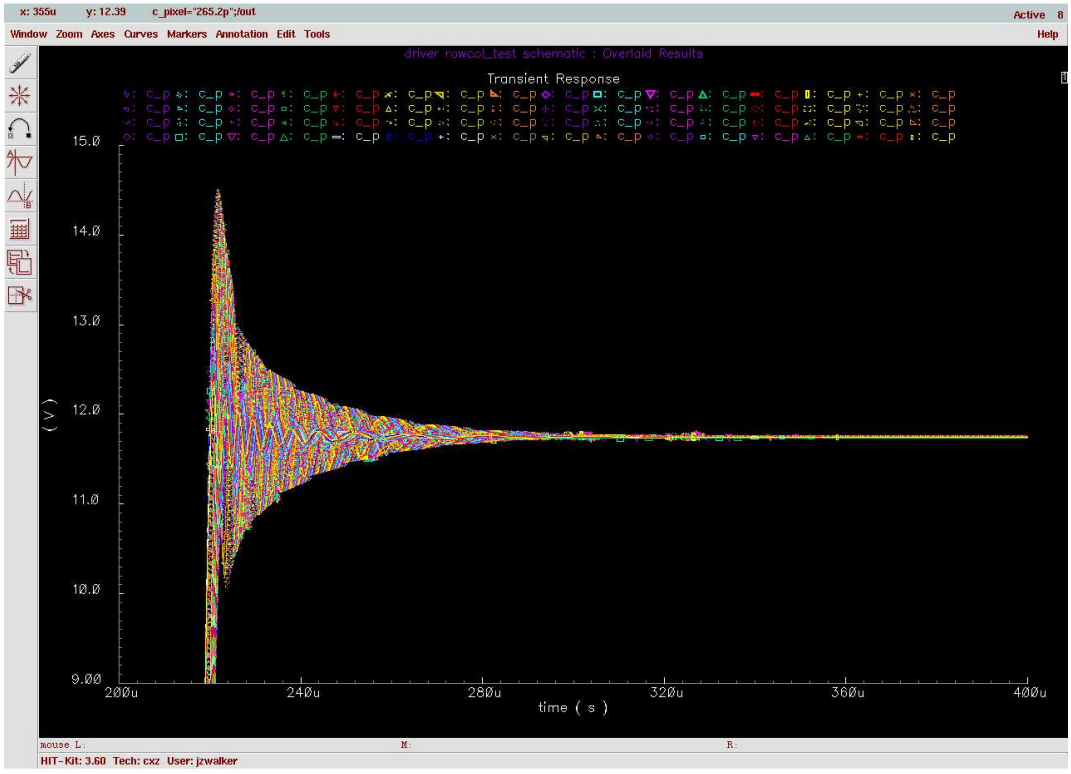

Figure A-1: Response over output load from 30p -> 400p

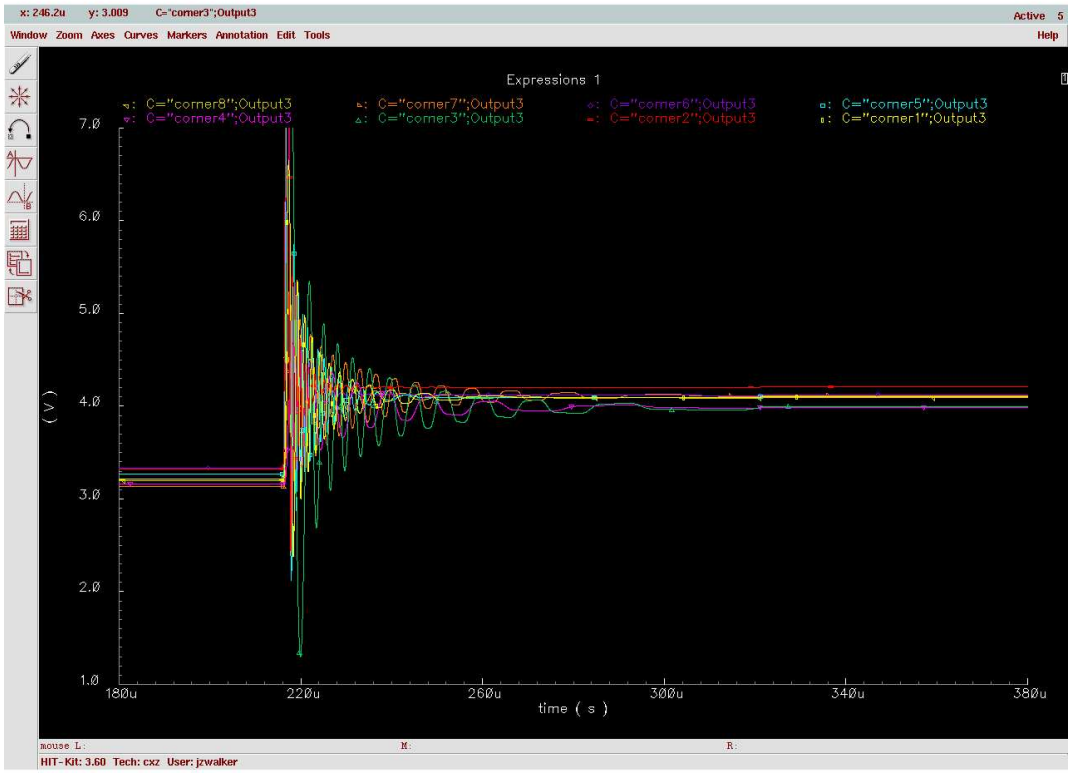

Figure A-2: Response over all corners (digital value 1)

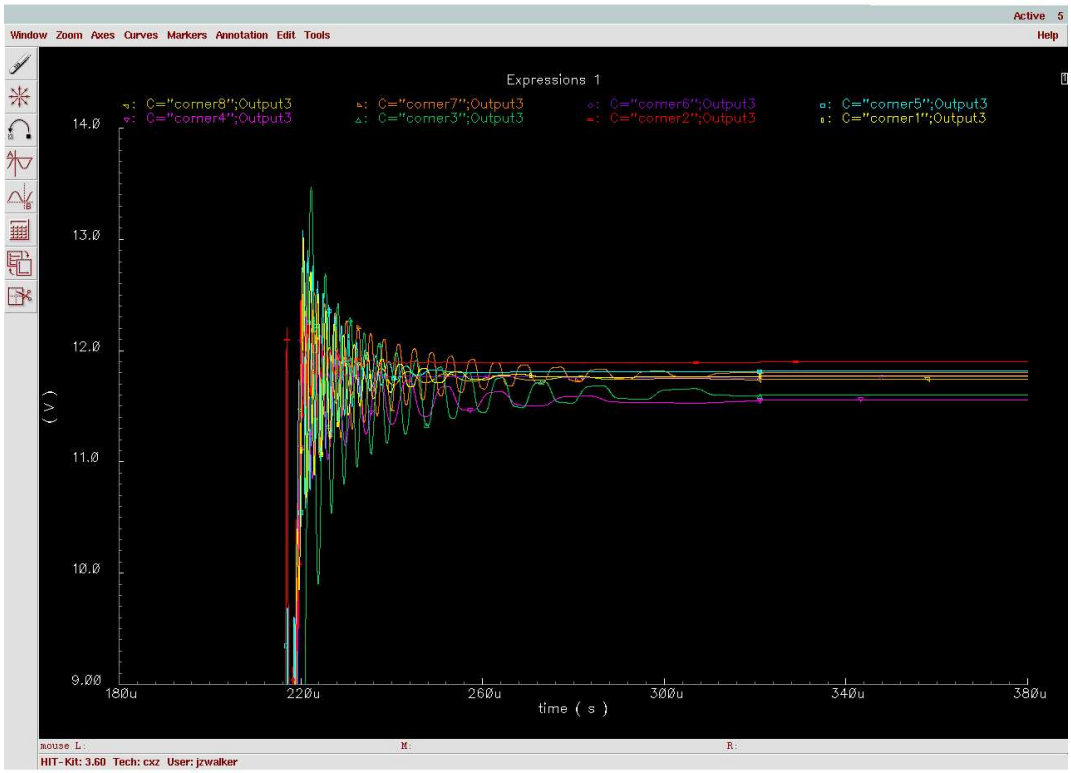

Figure A-3: Response over all corners (digital value 128)

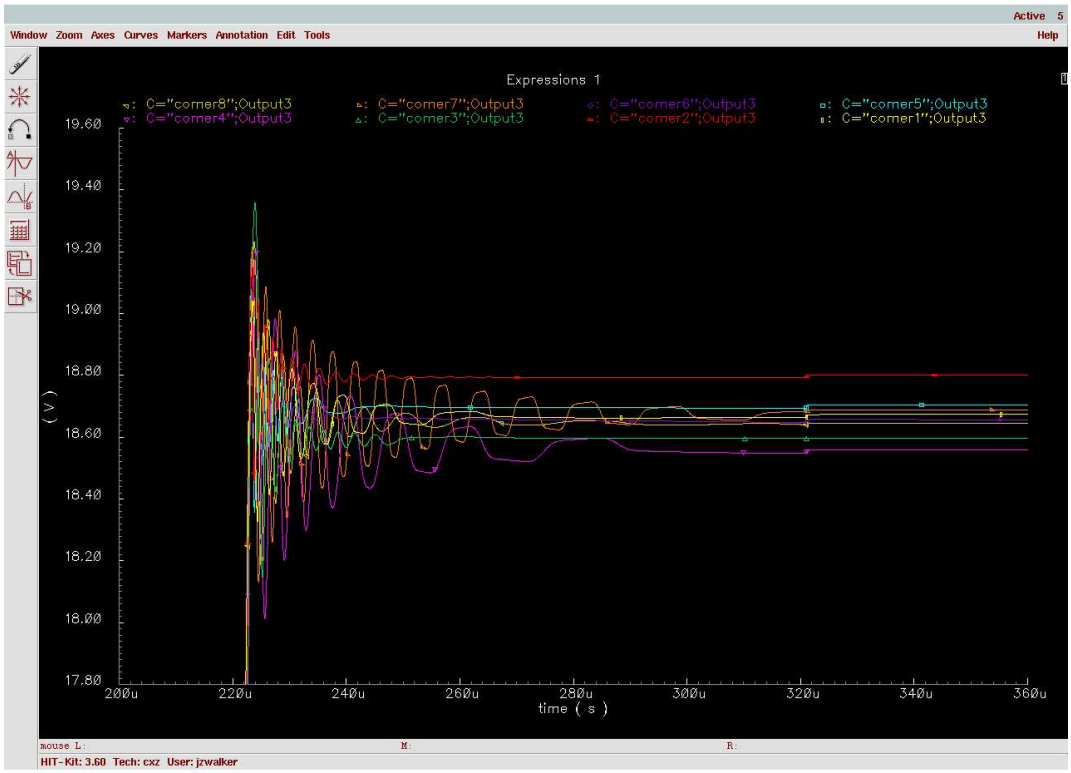

Figure A-4: Response over all corners (digital value 240)

Bibliography

[1] Michael Adams and Faouzi Kossentini. JasPer: A Software-Based JPEG-2000 Codec Implementation. *Proceedings of the International Conference on Image Processing*, pages 53–56, September 2000.

[2] Paul Alt and Peter Pleshko. Scanning Limitations of Liquid-Crystal Displays. *IEEE Transactions on Electron Devices*, pages 146–155, February 1974.

[3] A. Nakazawa et al. Ultra-low-power STN-LCDs using multiple-line addressing for mobile telecommunications applications. *Journal of the SID*, pages 277–280, 1999.

[4] Mohammed Ghanbari. *Video Coding: An Introduction to Standard Codecs*, chapter 3, pages 20–49. IEE Telecommunications Series 42. The Institution of Electrical Engineers, Herts, United Kingdom, 1999.

[5] Mohammed Ghanbari. *Video Coding: An Introduction to Standard Codecs*, chapter 2, page 19. IEE Telecommunications Series 42. The Institution of Electrical Engineers, Herts, United Kingdom, 1999.

[6] Supertex inc. *32-Channel 128-Level Amplitude Gray-Shade Display Column Driver*, internal/preliminary edition, August 2001.

[7] ISO/IEC. JPEG 2000 Part I Final Committee Draft Version 1.0. April 2000.

[8] Valencia Joyner. A low power display driver architecture for organic light emitting diode microdisplays. M. Eng. Thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 1999.

[9] M. Kitamura and Y. Hirai. Applications of Multiple Line Addressing (MLA) Technique and it's Architecutre. In *Proceedings of 1997 International Display Research Conference and Workshops*, pages M71–74. SID, September 1997.

[10] N.A. Lawrence, T.D. Wilkinson, and W.A. Crossland. Integrated Image Decompression and Display Driving for Portable Communication Devices Using DWTs. In *Proceedings of the SPIE - The International Society for Optical Engineering*, pages 197–208. SPIE, April 2001.

[11] N.A. Lawrence, T.D. Wilkinson, and W.A. Crossland. Experimental Verification of a Novel Multiple Line Addressing (MLA) Scheme for Passive Matrix LCDs. *Eurodisplay*, pages 955–958, 2002.

[12] Ernst Lueder. *Liquid Crystal Displays - Addressing Schemes and Electro-Optical Effects*, chapter 2, page 7. SID Series in Display Technology. John Wiley & Sons, Ltd., Chichester, England, 2001.

[13] Ernst Lueder. *Liquid Crystal Displays - Addressing Schemes and Electro-Optical Effects*, chapter 12, page 127. SID Series in Display Technology. John Wiley & Sons, Ltd., Chichester, England, 2001.

[14] Ernst Lueder. *Liquid Crystal Displays - Addressing Schemes and Electro-Optical Effects*, chapter 12, page 174. SID Series in Display Technology. John Wiley & Sons, Ltd., Chichester, England, 2001.

[15] Ernst Lueder. *Liquid Crystal Displays - Addressing Schemes and Electro-Optical Effects*, chapter 12, page 189. SID Series in Display Technology. John Wiley & Sons, Ltd., Chichester, England, 2001.

[16] Ernst Lueder. *Liquid Crystal Displays - Addressing Schemes and Electro-Optical Effects*, chapter 11, page 165. SID Series in Display Technology. John Wiley & Sons, Ltd., Chichester, England, 2001.

[17] Jurgen Nehring and Allan Kmetz. Ultimate Limits for Matrix Addressing of RMS-Responding Liquid-Crystal Displays. *IEEE Transactions on Electron Devices*, pages 795–802, May 1979.

[18] T.J. Scheffer and B. Clifton. Active Addressing. *Journal of the SID*, pages 94–104, September 1991.

[19] Je-Hurn Shieh, Mahesh Patil, and Bing Sheu. Measurement and Analysis of Charge Injection in MOS Analog Switches. *IEEE Journal of Solid State Circuits*, 22:277–281, April 1987.

[20] Amit Sinha and Anantha Chandrakasan. JouleTrack - A Web Based tool for Software Energy Profiling. In *Proceedings of the Design Automation Conference (DAC'01)*, pages 220–225. IEEE, June 2001.

[21] Kenneth Stafford, Paul Gray, and Richard Blanchard. A Complete Monolithic Sample/Hold Amplifier. *IEEE Journal of Solid State Circuits*, SC-9 (No. 6):381–387, December 1974.

[22] Gilbert Strang and Truong Nguyen. *Wavelets and Filter Banks*, section 2.5, page 71. Wellesley-Cambridge Press, Wellesley, Massachusetts, 1996.

[23] Ricardo Suarez, Paul Gray, and David Hodges. All-MOS Charge Redistribution Analog-to-Digital Conversion Techniques–Part II. *IEEE Journal of Solid State Circuits*, 10:379–385, December 1975.

[24] Wim Sweldens. Wavelets and the lifting scheme: A 5 minute tour. *Magazine for Applied Mathematics and Mechanics*, 76 (Suppl. 2):41–44, 1996.

[25] T.Kurumisawa, A. Ito, S. Yamazaki, and S. Iino. High-Performance Ultra-Low-Power 640x200 Reflective STN Display System Using Four-Line Selection Method. *Journal of the SID*, pages 351–354, 1996.

[26] Stephen Welstead. *Fractal and Wavelet Image Compression Techniques*, section 5.8, page 111. Tutorial Texts Series. The International Society for Optical Engineering, Bellingham, Washington, 1999.

[27] Chih wen Lu and Chung Lee. A Low Power High Speed Class-B Buffer Amplifier for Flat Panel Display Application. In *Proceedings of the First IEEE International Workshop on Electronic Design, Test and Applications (DELTA'02)*, pages 172–176. IEEE, January 2002.