SOFTWARE BASED NTSC SIGNAL PROCESSING

A Thesis submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Technology

by

Prateek Mohan Dayal
Roll No. 01010217

under the guidance of

Dr. Debashis Ghosh

Department of Electronics and Communication Engineering
Indian Institute of Technology, Guwahati

May 2005

Certificate

This is to certify that the work titled **"Software Based NTSC Signal Processing"** has been done by Prateek Mohan Dayal under my supervision and it has not been submitted elsewhere for a degree.

Dr. Debashis Ghosh
Assistant Professor
Department of ECE
IIT Guwahati
India

Abstract

Software Defined Radio is a new design paradigm which is trying to move the code closer to the antenna in a communication system. Several commercial and open source projects are lending the power of software to traditionally hardware oriented communication tasks. This thesis presents the results of our two semester long work done as a part of the B.Tech Project titled "Software Based NTSC Signal Processing". It begins by introducing the concept of software radio and initiates the reader to GNU Radio, an open source tool for creating software applications. Design approach of GNU Radio is exemplified through NTSC audio decoding application. The report ends with a discussion on the design and implementation of monochrome and color NTSC video decoding and provides directions for future work.

Acknowledgments

First of all I would like to thank my parents for their constant love and encouragement. I am extremely grateful to my project supervisor, Dr. Debashis Ghosh for giving me the independence of proposing my own project topic and for constantly motivating me to be creative with my work. I am extremely thankful to Dr. U.B. Desai, Professor, Electrical Engineering Department, IIT Bombay for exposing me to wonderful research opportunities in the area of Signal Processing for Communications. I also wish to thank Prof. Anil Mahanta of IIT Guwahati for the fruitful discussions on signal processing that I have had with him over the last two years. I am very thankful to the entire GNU Radio community, specially Eric Blossom for their constant help. Finally I would like to thank all my friends at IIT Guwahati for leaving me with just about enough time to finish this work !!!

To Hardware/Software Open Source Community ...

Contents

List of Figures

Chapter 1

Introduction

1.1 Fundamentals of Software Radio

A software radio performs the basic transmitter and receiver actions (modulation and demodulation) in software. At the transmitter end, waveforms are generated in software as samples and then a Digital to Analog Converter (DAC) converts the waveform into an analog waveform that can be upconverted if required and then transmitted. At the receiving end, the downconverted Intermediate Frequency (IF) waveform is sampled and the samples are then analyzed by a code. The term *Software Radio* was coined by Joseph Mitola in 1991. A more recent concept is Cognitive Radios, referring to a class of software radios that employ several complex algorithms in planning and learning radio etiquettes. Software Radios offer greater flexibility in implementation and reconfigurability. This is because the programming environment for a Personal Computer (PC) is much more general than the programming environment for say Digital Signal Processing (DSP) kits, thereby allowing better code reuse and in turn reduced development time. Also more than one standard can be implemented on a single system with the same hardware being used for a multitude of applications. Moreover with increasing processing speeds the power of software radios is expected to grow over time, justifying the research thrust in this area.

1.2 GNU Radio

Apart from commercial software radio platforms available from companies such as Vanu Inc [1], open source softwares such as GNU Radio [2] are creating waves in the software radio field these days. The GNU Radio project [2] was initiated and is currently maintained by Eric Blossom with contributions from a worldwide community of developers. In essence, GNU Radio is a collection of open source software that lets a linux based PC construct and analyze radio waveforms with some basic radio frequency (R.F) hardware and ADC/DAC kits. It is more popular in the open source community as a tool that lets one "hack" the electromagnetic spectrum.

GNU Radio provides a library of signal processing blocks and the glue to tie it all together. The programmer builds a radio by creating a graph (as in graph theory) where the vertices are signal processing blocks and the edges represent the data flow between them. The signal processing blocks are implemented in C++. Conceptually, blocks process infinite streams of data flowing from their input ports to their output ports. Blocks' attributes include the number of input and output ports they have as well as the type of data that flows through each. The most frequently used types are short, float and complex. Some blocks have only output ports or input ports. These serve as data sources and sinks in the graph. There are sources that read from a file or ADC, and sinks that write to a file, digital-to-analog converter (DAC) or graphical display. About 100 blocks come with GNU Radio. Writing new blocks is not difficult.

On the hardware side, the requirements include an RF frontend, like a cable tuner module and a sampler, like Measurement Computing PCI DAS card or a Universal Software Radio Platform (USRP) [3], a sound card for listening to the sound output and in some cases also to serve as a narrow band ADC for signal input (example AM reception). Fig. 1.1 shows a USRP device. GNU Radio is currently available on Linux platform and apart from the GNU Radio core files, there is a driver for PCI DAS card, USRP kit, 4397 cable tuner module. Apart from working implementations of FM, HDTV, AM, projects under investigation or in progress include a TiVo equivalent for radio, capable of recording multiple stations simultaneously, Radio astronomy, Software GPS, Ad hoc mesh networks etc.

1.3 BTP: Motivation and Objective

Although as mentioned above, GNU Radio supported with the USRP kit is capable of receiving multiple FM stations simultaneously in realtime, HDTV and AM transmission, NTSC/PAL decoding is still not available in GNU Radio. This has motivated our current research goal.

In this project, we study the GNU Radio platform and develop NTSC decoding software based on it. Our aim is to decode NTSC composite video and audio signal and use the computer monitor with the sound card as output device. Software demodulation of analog TV signal offers the following benefits

- The RF frontend, USRP etc in this case, are general devices and can be used for other applications as well. Only component specific to NTSC decoding is the code.

- Depending upon the computational resources available to the system, the receiver performance can be improved. For example if the system resources are available, the receiver can switch from a 2 line comb filter to 3 line comb filter or even an adaptive comb filter.

1.4 Organization of the thesis

Chapter two describes the signal acquisition setup and the mathematics involved in it. Chapter three discusses audio decoding under the GNU Radio framework. Chapter four and five discuss composite video signal structure and decoding. Chapter six has been devoted to color decoding. Chapter seven discusses the experimental results and discusses several implementation issues associated with the receiver implementation. Conclusions are presented in chapter eight along with the directions for future work in chapter nine.

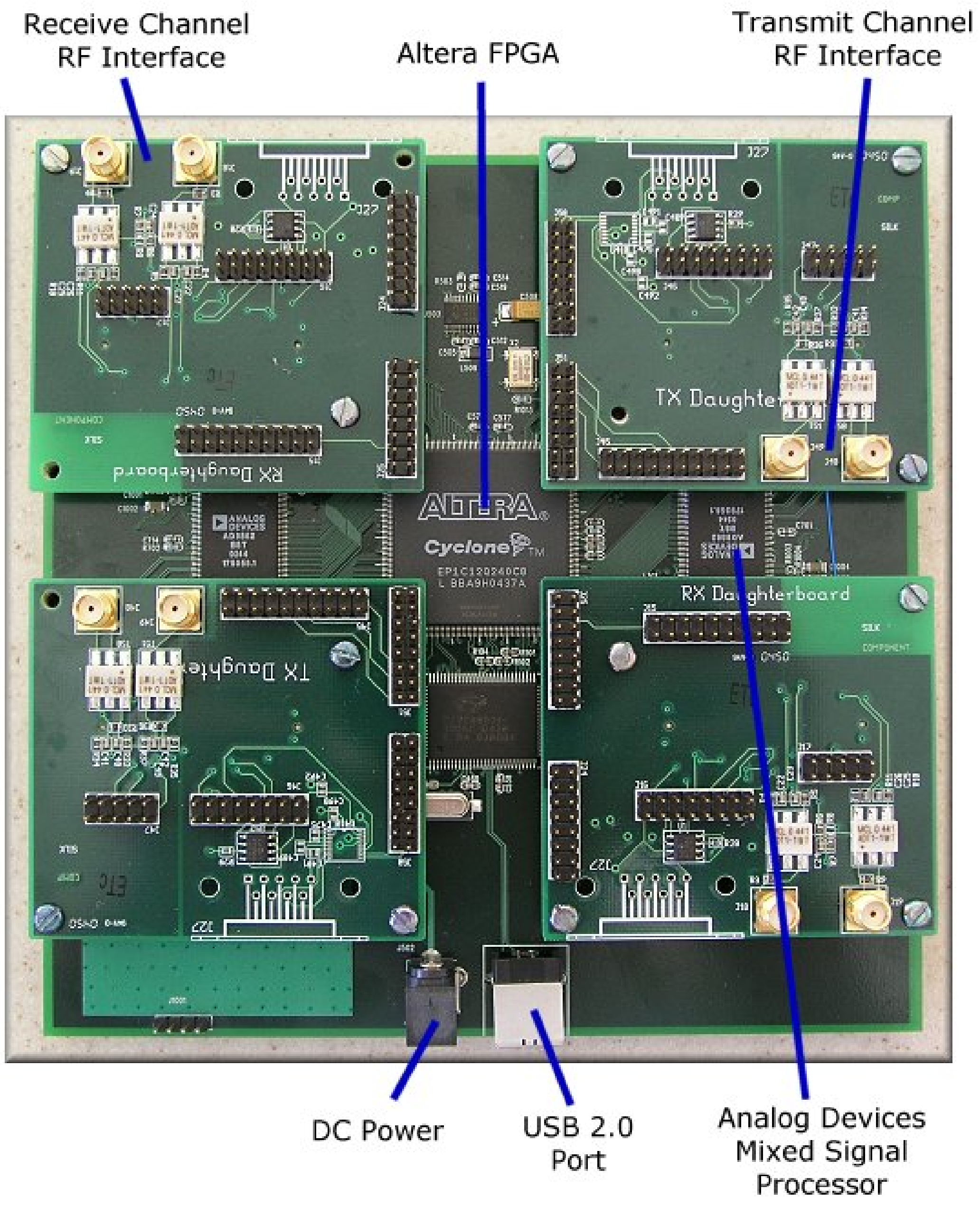

Receive Channel RF Interface
Altera FPGA
Transmit Channel RF Interface
DC Power
USB 2.0 Port
Analog Devices Mixed Signal Processor

Figure 1.1: Universal Software Radio Platform

Chapter 2

Signal Acquisition and Structure

Signal acquisition is one of the most critical and difficult part of the whole application. This chapter throws light on the hardware setup used for capturing over the air (OTA) NTSC signal and discusses the mathematical steps involved in quadrature sampling the signal [4].

2.1 Hardware Setup

TV signal is a VSB modulated signal with signal bandwidth of about 6 MHz for NTSC and 7 MHz for PAL broadcast standard. Therefore after downconversion, we need a sampler that can sample at a rate of atleast 12 MSamples/sec. This rate can be significantly lowered if we quadrature sample the signal with its spectrum centred at 0 Hz. This way we generate two data streams, an Inphase and a Quadrature stream constituting complex samples.

The preferred way to acquire signal for further processing in GNU Radio is to use USRP. Current version of the USRP features 4 High-Speed AD Converters (64 MS/s, 12-bit Analog Devices AD9862) which can bandpass-sample signals of up to about 200 MHz, digitizing a band as wide as about 32 MHz, 4 High-Speed DA Converters (128 MS/s, 14-bit) to generate signals up to about 50 MHz. The interface for the USRP kit is high speed USB 2.0 which is fast enough for sustaining 32 MBps to a PC. One of the main features of the USRP is the presence of Altera EP1C12 Q240C8 "Cyclone" Field Programmable Gate Array (FPGA) for high bandwidth math. This FPGA is also used for "decimation" so that the final output is at a rate that can be transferred to the PC over the USB interface. The usual I/O format is 16-bit I and 16-bit Q data (complex), resulting in 8M complex samples/sec across the USB.

The RF frontend used between the antenna and the A/D chip onboard the USRP is a Microtune 4937 cable modem tuner module [5]. The tuner module can downconvert frequencies ranging from 50 MHz to 860 MHz to an I.F of 43.75 MHz (US) or 36.125 MHz (Europe). There is a second downconversion step available which outputs to 5.75 MHz (US) or 7.125 MHz (Europe). In the setup used to acquire NTSC samples for this project, tuner output was taken from the second stage at an I.F frequency of 5.75 MHz. The module also contains an I.F amplifier so that we can directly sample the output at this stage. The tuner module can be controlled over the parallel port with the control software for linux available at any of the GNU Radio mirrors. It should be noted that it is the centre of the 6 MHz band that is at 5.75 MHz and not the picture carrier. The picture carrier is at 3.75 MHz and the audio carrier is at 8.25 MHz.

Output of the cable tuner is fed to one of the onboard ADC. The I.F signal at 5.75 MHz is sampled at 64 MSamples per second, 12 bits/sample and then later on downconverted to complex baseband and decimated to 8M samples per second in the USRP itself. The data thus contains real and imaginary 16 bit numbers. The theory of quadrature sampling is used for this purpose, which allows us to sample at much lower sampling rate if we use two samplers that result in I and Q channel or the real and imaginary channel. The next section describes the mathematics involved in this step.

2.2 Quadrature Sampling Steps

The picture component of the composite signal can be written as

$$s(t) = A_c \left[1 + \frac{1}{2}k_a m(t)\right] cos(2\pi f_c t) + \frac{1}{2}A_c k_a m^{'}(t) sin(2\pi f_c t) \tag{2.1}$$

where f_c is the picture carrier frequency (3.75 MHz at the output of the tuner module), $m(t)$ is the composite video signal consisting of sync information, luminance information and chrominance information. As explained in a later chapter, chrominance information is DSBSC modulated using a chroma subcarrier of 3.58 MHz. In the above expression $m^{'}(t)$ is the output of the VSB filter with $m(t)$ as input [6].

The following steps will bring down the centre of the single sided video spectrum to baseband and give us the complex analytic time domain samples of the NTSC broadcast signal. Let f_{LO} be the frequency of the local oscillator and θ_{LO} be the initial phase of the local oscillator's inphase waveform. Then the inphase component of the downconverted signal, before lowpass filtering can be written as

$$\begin{aligned} i^{'}(t) &= s(t)cos(2\pi f_{LO}t + \theta) \\ i^{'}(t) &= A_c \left[1 + \frac{1}{2}k_a m(t)\right] cos(2\pi f_c t)cos(2\pi f_{LO}t + \theta) \\ &+ \frac{1}{2}A_c k_a m^{'}(t) sin(2\pi f_c t)cos(2\pi f_{LO}t + \theta) \end{aligned} \tag{2.2}$$

Since f_{LO} is comparable in order to f_c, the above signal after low pass filtering can be represented as

$$\begin{aligned} i(t) = &\frac{1}{2}A_c\left[1+\frac{1}{2}k_am(t)\right]cos\left(2\pi(f_c-f_{LO})t-\theta)\right) \\ &+\frac{1}{4}A_ck_am^{'}(t)sin\left(2\pi(f_c-f_{LO})t-\theta\right) \end{aligned} \tag{2.3}$$

Similarly the quadrature channel output can be written as

$$\begin{aligned} q(t) = &-\frac{1}{2}A_c\left[1+\frac{1}{2}k_am(t)\right]sin(2\pi(f_c-f_{LO})t-\theta) \\ &+\frac{1}{4}A_ck_am^{'}(t)cos(2\pi(f_c-f_{LO})t-\theta) \end{aligned} \tag{2.4}$$

If $f_c - f_{LO} = -1.75$ MHz and $\theta = 0$, then the 5.75 MHz I.F is shifted to 0 MHz. The audio carrier is shifted to 2.75 MHz, thereby limiting the spectrum to well within (-4,4) MHz. Therefore each of the inphase and quadrature components can be sampled at 8 MHz at a resolution of 16 bits each, giving an effective data rate of 32 MBps. All these steps are programmed in the USRP and the final data samples are transferred to the PC over the USB 2.0 interface.

Fig.2.1 and Fig.2.2 show the sampler structure and action graphically. When $f_c - f_{LO} = 0$ and $\theta = 0$, the complex samples, scaled by a factor of two can be written as

$$s_{cmp}[n] = A_c\left[1+\frac{1}{2}k_am(n)\right]+\frac{1}{2}A_ck_am^{'}(n)j \tag{2.5}$$

Fig.2.3 shows the spectrum of the discrete time complex signal given by (2.5) with picture carrier at 0 Hz.

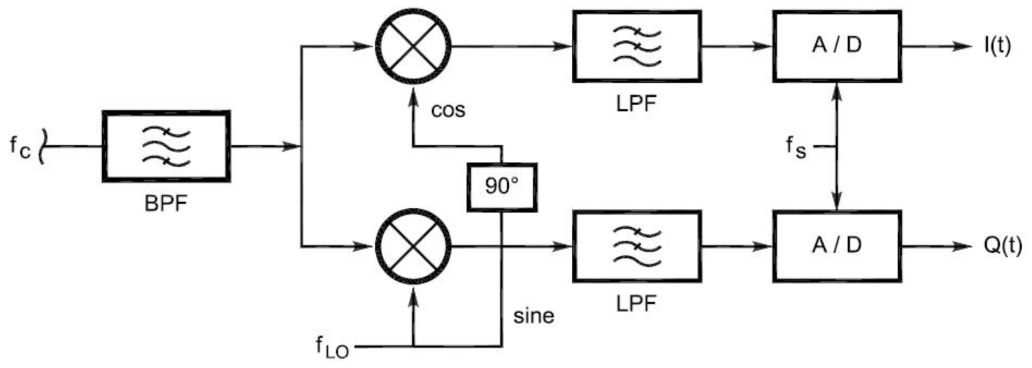
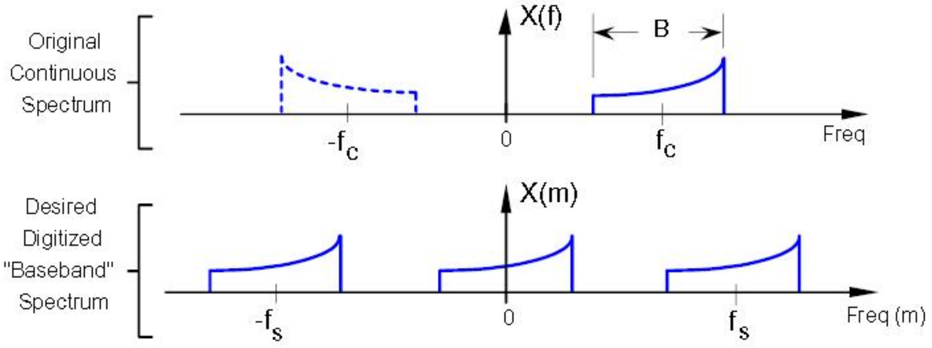


Figure 2.1: Quadrature Sampler Structure



Original Continuous Spectrum
X(f)
B
$-f_c$
0
f_c
Freq
Desired Digitized "Baseband" Spectrum
X(m)
$-f_s$
0
f_s
Freq (m)

Figure 2.2: Quadrature Sampling Objective

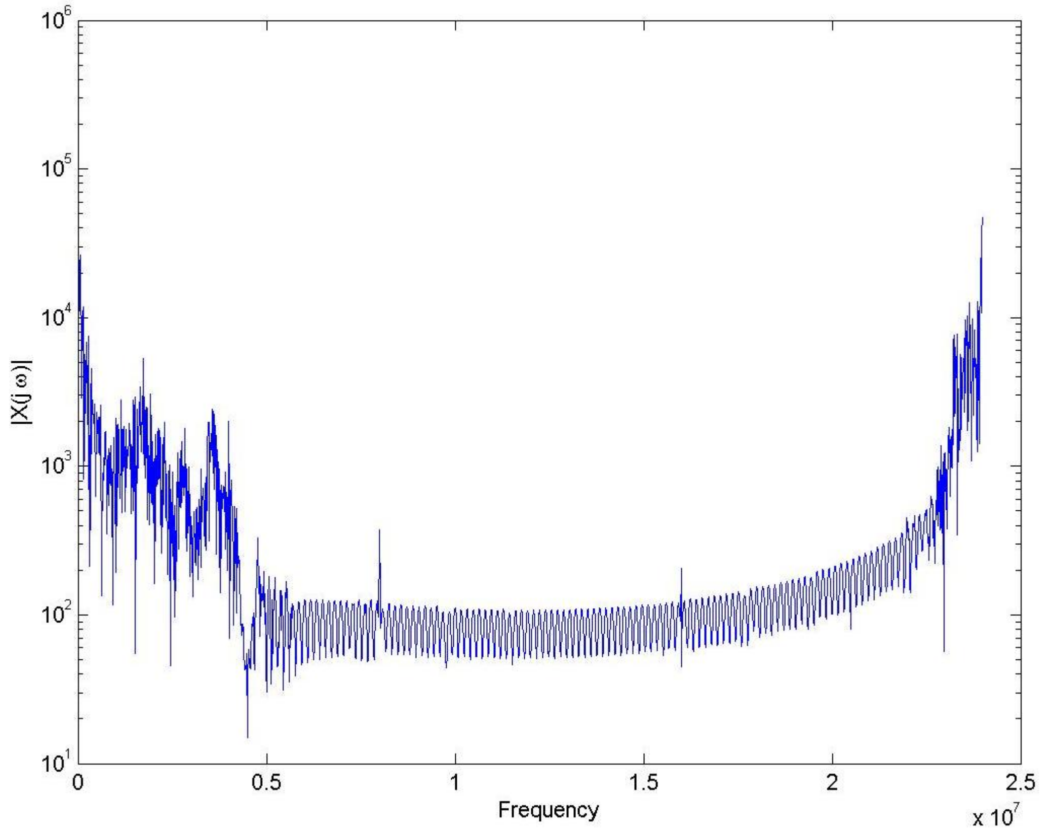


Figure 2.3: FFT of the NTSC signal with Picture Carrier at 0 Hz

Chapter 3

Audio Signal Structure and Decoding

3.1 Structure of the Audio Signal

Audio signal in NTSC standard is frequency modulated. Audio information is encoded as frequency deviation around the carrier frequency f_c, which happens to be 4.5 MHz in NTSC. Considering a sinusoidal modulating signal, $m(t) = A_m sin(2\pi f_m t)$, the corresponding FM signal can be written as

$$s(t) = A_c cos\left[2\pi f_c t + \frac{k_f A_m}{f_m} sin(2\pi f_m t)\right] \tag{3.1}$$

where k_f is a constant of proportionality.

3.2 Quadrature FM Demodulator

If the FM signal given by (3.1) is downconverted to baseband and quadrature sampled, the corresponding complex samples can be written as

$$\begin{aligned} s_{cmp}[n] &= A_c cos(\beta sin(2\pi \frac{f_m}{f_s} n)) + A_c sin(\beta sin(2\pi \frac{f_m}{f_s} n))j \\ &= A_c e^{j\beta sin(2\pi \frac{f_m}{f_s} n)} \end{aligned} \tag{3.2}$$

where f_s is the sampling frequency and $\beta = \frac{k_f A_m}{f_m}$. Product of the n^{th} and $n-1^{th}$ sample's complex conjugate is given by

$$\begin{aligned} s_{cmp}(n) s^*_{cmp}(n-1) &= A^2_c e^{\left\{2j\left[2\pi\left(\frac{f_m}{f_s} n - \frac{f_m}{f_s}\right)\right] sin\left(\frac{\pi f_m}{f_s}\right)\right\}} \\ &= A^2_c e^{\left\{2j\beta sin\left[2\pi\left(\frac{f_m}{f_s} n - \frac{f_m}{f_s}\right)\right] \frac{\pi f_m}{f_s}\right\}} \end{aligned} \tag{3.3}$$

phase of the above product is proportional to the message signal $m(t)$, phase shifted by $2\pi \frac{f_m}{f_s}$. This approach to FM demodulation is an example of the power of quadrature processing applied to DSP.

3.3 Signal Flow Graph for FM decoding

As discussed earlier in the chapter on GNU Radio, every application in GNU Radio is modeled as a signal flow graph starting from a source and ending in a sink, generally sound card or file. Fig.3.1 shows the signal flow graph for decoding FM. The source in this case is the file containing the complex samples and the sink is the soundcard. Decimation step is required to reduce the computational load wherever possible. In this application, after downconversion, the band of interest (FM) is only about 200 kHz wide and therefore a sampling rate of 400 kHz is enough beyond downconversion if we take care of the aliasing . This way we can reduce the computational complexity by a factor of 20. This graph can be constructed and manipulated during runtime in python [7]. It calls the core signal processing blocks implemented in C++. This is made possible by Simplified Wrapper and Interface Generator software package [8].

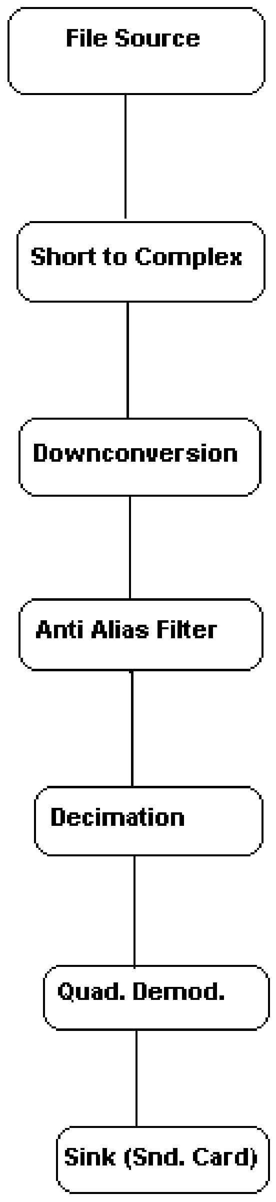


Figure 3.1: Signal Flow Graph for FM decoding

Chapter 4

Composite Video Signal Structure

Composite video signal consists of luminance, chrominance and sync information required to build a complete color frame. As described later in the thesis, color signal is compatible with monochrome televisions, that use luminance information along with the timing information to generate and display monochrome frames.

NTSC system uses 30 frames per second, each frame consisting of an even and odd field. Each field consists of 262.5 scan lines, a total of 525 lines per frame [9]. Out of the 525 lines, only about 480 lines contain active picture information. Other lines are used for vertical blanking or for text captions. Fig.4.2 shows important amplitude characteristics of NTSC signal.

4.1 Timing Information

Horizontal and Vertical blanking period indicate the beginning of a new scan line and field respectively. The sync pulses constitute the uppermost 25% of the composite video and lie in a region which is blacker than black.

4.1.1 Horizontal Blanking Period

Fig.4.3 shows the $10.2\mu sec$ horizontal blanking period zoomed in. Front porch of duration $1.26\mu sec$ is present to give enough cushion to the electron gun to switch off in case a very bright line is currently being displayed. Duration of the horizontal sync pulse is $4.75\mu sec$ with an error margin of $0.5\mu sec$. Duration of the back porch of is guaranteed to be atleast $3.81\mu sec$. Back porch contains the color burst signal, a section of color subcarrier otherwise suppressed but reinserted here to aid in synchronous decoding of chroma signal.

4.1.2 Vertical Blanking Period

Fig.4.4 shows the vertical blanking period zoomed in. There are six serrated pulses of total duration .8 to 1.3 msec. Also present before and after the vertical sync pulses are the six pre-equalizing and six post equalizing pulses, each of duration $2.54\mu sec$ separated from each other by a distance $\frac{H}{2}$. H here is the duration of a horizontal scan line, equal to $63.5\mu sec$. Purpose of the pre and post equalizing is to help the integrator circuits in TV sets trigger at the same instant for both the fields. In the proposed monochrome demodulator, location of the first pre-equalizing pulse decide if the field in question is even or odd. This information is of use when combining the even and odd fields to reconstruct the deinterleaved frame.

4.2 Picture Information

In a horizontal scan line, about 52 to 54 μsec of signal after the horizontal blanking period is used for transmitting the picture information. This consists of both the monochrome and color information.

4.2.1 Luminance Information

The luminance information is used for both monochrome and color video decoding. Luminance information is negative amplitude modulated, with an increasing voltage level corresponding to a decrease in intensity. Owing to the periodicity of horizontal blanking pulses, in frequency domain the luminance information occurs in bundles of energy centred at even harmonics of the horizontal line frequency.

4.2.2 Chrominance Information

Chrominance modulation consists of modulating an inphase and a quadrature signal, each derived from RGB information onto a single carrier. This is accomplished using DSBSC modulation, with inphase and quadrature components determining amplitude and phase of the carrier signal. DSBSC demodulator requires an estimate of phase of the unmodulated carrier which it derives from the color burst signal located on back porch of the horizontal blanking period. Color subcarrier frequency is chosen to be $\frac{455}{2}f_H$, where f_H is the horizontal scan rate. This comes out to be about 3579545 Hz or 3.58 MHz approximately. Therefore the chrominance spectral energy occurs in bundles of energy centred at odd harmonics of the horizontal line frequency, as shown in Fig.4.1. In NTSC standard, the chrominance spectral components extend from 2.58 MHz to 4.05 MHz, however larger bandwidth of the I Channel is seldom used by TV receivers [10].

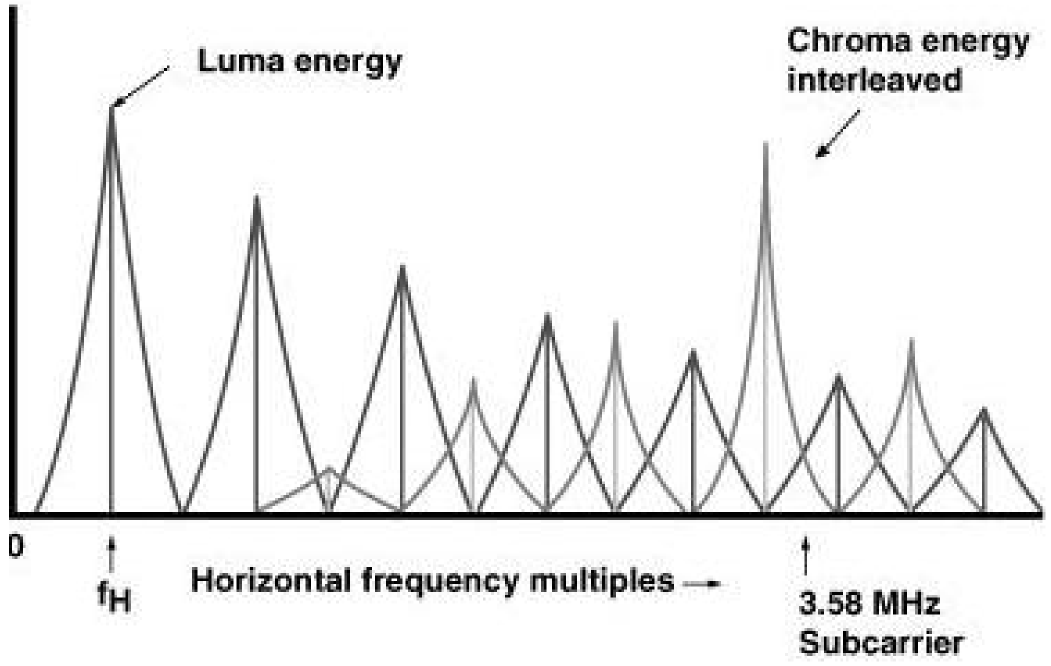


Figure 4.1: Interleaved Y/C Information

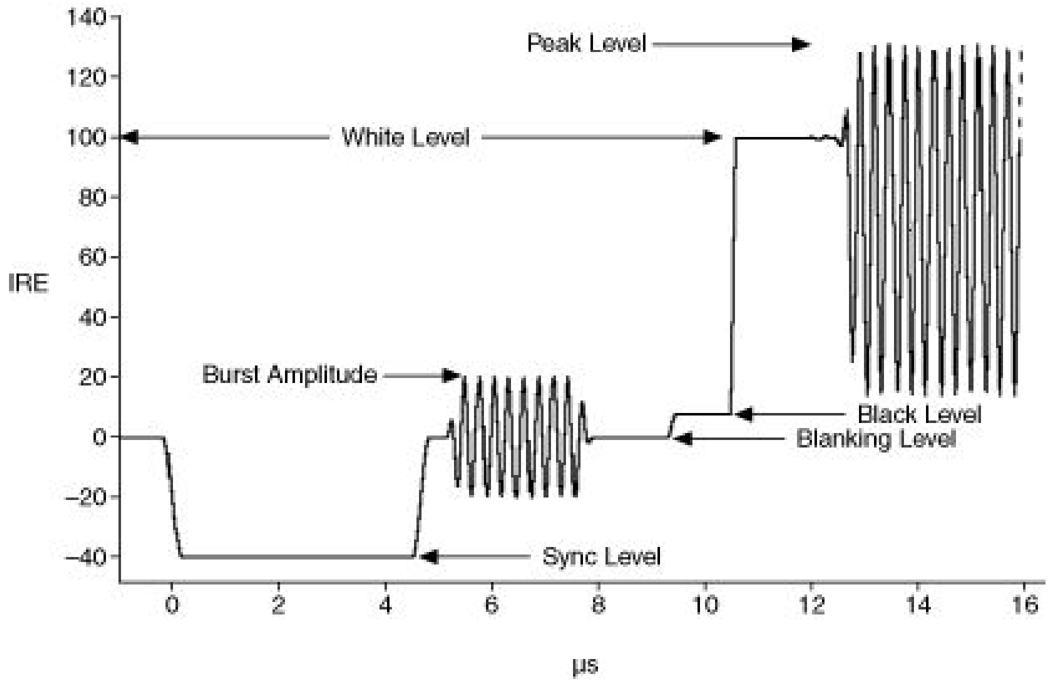


Figure 4.2: NTSC signal amplitude levels

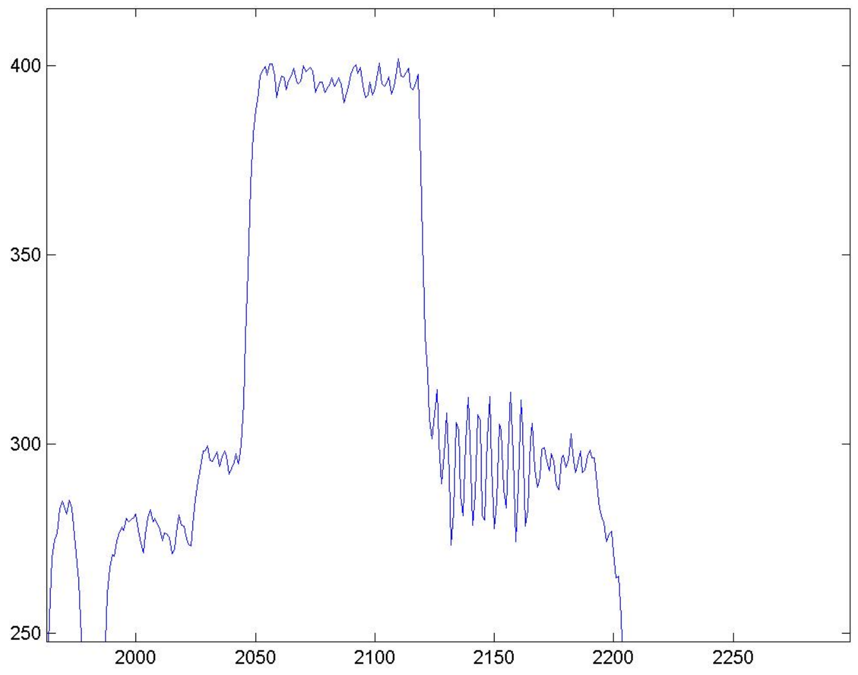


Figure 4.3: Horizontal Blanking Pulse

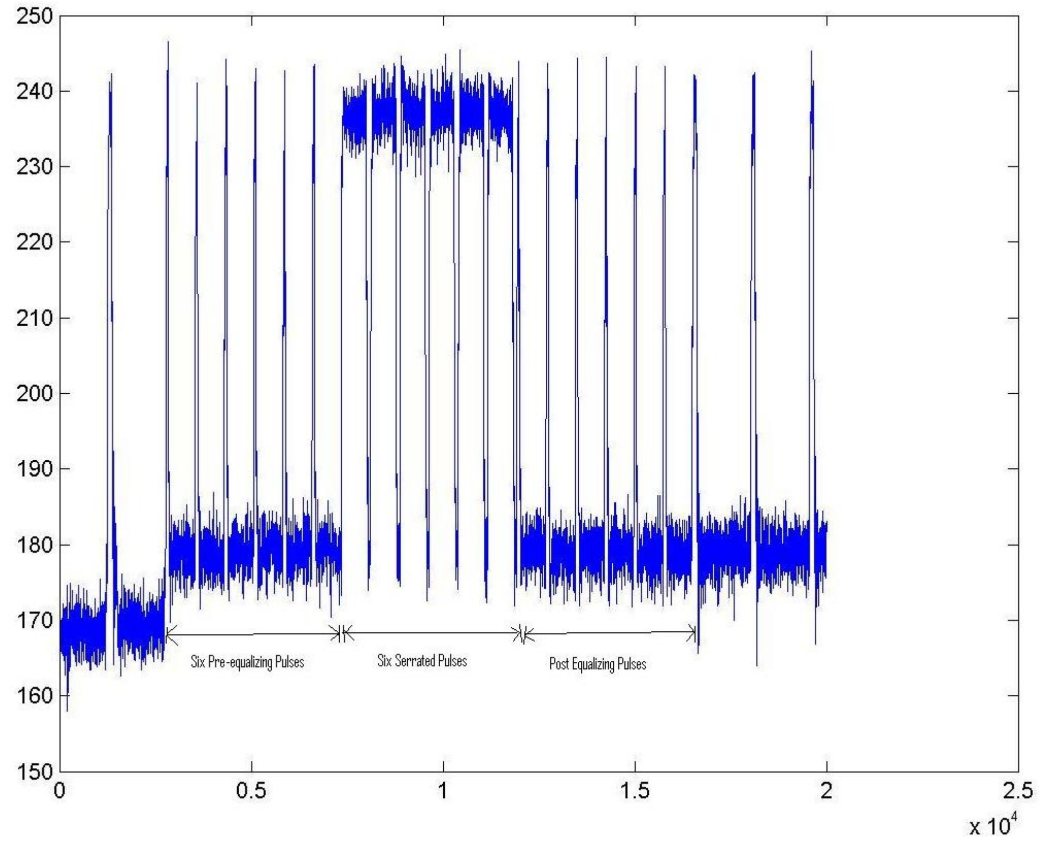


Figure 4.4: Vertical Blanking Pulse

Chapter 5

Monochrome Decoding

This chapter discusses the design and implementation issues of the proposed monochrome receiver. Receiver algorithms have currently been implemented and tested in Matlab. Complete implementation is available on the web [11]. Sections below describe the steps involved in generating a complete deinterleaved frame from the complex NTSC samples available offline in a file.

5.1 Interpolation and Downconversion

Even though the envelope detection required to decode luminance information can be performed with the picture carrier not at baseband, the picture carrier needs to be shifted to 0 Hz, so that the comb filter described in a later section can perform normally. Since the sampling frequency is 8 MHz, the spectrum above 4 MHz will get aliased to frequencies between -4 and 0 MHz. Therefore when performing downconversion, the audio carrier and some of the chrominance information will suffer aliasing. This effect is visible in Fig.5.2. We therefore increase the data rate by an order of three, resulting in a data rate of 24 MHz. This way, even after envelope detection, none of the spectral components undergo aliasing. Like any other interpolation exercise, the overall performance of interpolation step depends on the attenuation of the stopband of this filter.

5.2 Envelope Detection

As described in the previous sections, the timing information and luminance information is available as the envelope of the NTSC signal. A simple way to recover this information is to use an envelope detector. In circuit, an envelope detector is build using a diode detector followed by low pass filtering. Operation of the diode detector is modelled as squaring and this operation can be accomplished in software by taking the magnitude of complex samples. Magnitude of signal given by (2.5) can be written as

$$a[n] = A_c \left[1 + \frac{1}{2}k_a m[n]\right] \left\{1 + \left[\frac{\frac{1}{2}k_a m^{'}[n]}{1 + \frac{1}{2}k_a m[n]}\right]^2\right\}^{\frac{1}{2}} \tag{5.1}$$

which is proportional to the message signal but with some distortion contributed by $m^{'}[n]$. This distortion can be kept small if amplitude sensitivity k_a and vestigial sideband bandwidth are kept small. Width of the vestige in commercial TV systems is decided keeping these facts in mind. Even though the envelope information can be straight away used to draw the frame, it also contains chrominance information other than luminance information, leading to an effect known as cross luminance. To separate the chrominance information, we need to use a comb filter, but to implement it, accurate timing information is required. Next section describes how to achieve this.

5.3 Sync Processing

While sync separation can be achieved very simply by using a comparator, locating horizontal and vertical syncs is a bit more complex. Conventional

TV sets distinguish between the horizontal and vertical blanking based on their frequency content. Compared to the horizontal sync, vertical sync represents a very low frequency signal. Sync processing approach in this project is based on the difference in the duration of the vertical sync pulses and horizontal sync pulses. Based on this information, simple correlation based tests have been designed to identify horizontal and vertical syncs and their location.

5.3.1 Decimation

The interpolation step was previously performed to make sure that there is no aliasing in the subsequent downconversion and envelope detection operations. However, the envelope for sync processing need not contain very high frequency spectral components and therefore can be decimated to the original rate of 8 MSamples/sec after low pass filtering with a cutoff frequency of 3 MHz. This operation has the effect of reducing the computational complexity of the subsequent sync processing operations by a factor of 3. The corresponding locations in the interpolated stream can be estimated from the values in the decimated stream by a multiplication factor of 3, as in both interpolation and decimation these sample values are precisely the same.

5.3.2 Vertical Sync Processing

First step in sync processing is to identify the vertical sync. This step is computationally the most expensive in all the sync processing steps. Correlation sequence used for identifying a vertical sync pulse is shown in Fig.5.3. Initially the search needs to be performed in a sample size equal to a vertical line duration, i.e., about 130 Ksamples. Once the first vertical sync pulse has been identified, location of the next vertical sync pulse is more or less fixed. However there may be slight deviation from the expected value, which can be corrected by a search in a very small sample size. Therefore the computational complexity is greatly reduced after the first vertical sync detection. Fig.5.4 shows the output of the correlation performed to identify the first vertical sync pulse. The lag corresponding to the peak value determines the starting of the vertical sync pulse.

5.3.3 Horizontal Sync Processing

Once a vertical sync pulse has been identified, the system knows where to find the horizontal sync pulse. Therefore, we can perform the search for a horizontal pulse in a very small sample size using the correlation sequence shown in Fig.5.5. Assuming that the horizontal sync pulse is 4.75 microsecond in the ideal case, we can calculate, based on the length of the correlation output, jitter in the sync pulse and compensate for it accordingly later in the code. Fig.5.6 shows the correlation output for horizontal sync search.

5.3.4 Even/Odd Field Detection

To build a complete deinterleaved frame from the decoded information, the receiver needs to know if the field it is currently working on is even or odd. Based on this information, it knows what scan lines to skip after the vertical sync pulses and how to construct the deinterleaved frame. Even/Odd field detection can be performed based on the distance of the last horizontal sync pulse before starting of the pre-equalizing pulses and the first pre-equalizing pulse. If this distance is H, the field is odd whereas if this distance is $\frac{H}{2}$, the field is even.

Fig.5.7 shows the even field and Fig.5.8 shows the odd field. Fig.5.9 shows the resulting de-interleaved monochrome frame before Y/C separation.

5.4 Y/C Separation

Several approaches to Y/C separation are discussed below

5.4.1 Lowpass and Bandpass Filtering

Conventionally Y/C separation has been achieved by low pass filtering for luminance and bandpass filtering for chrominance information. The lowpass filter cutoff is generally taken to be about 3 MHz. However this approach results in the leakage of certain chrominance information into the luminance information and vice versa, leading to cross luminance and cross chrominance effects. Also due to the loss of high frequency information, the monochrome frame may lose sharpness.

5.4.2 Comb Filtering

Theory of comb filtering was know long back but their implementation has become easy and affordable only recently. Action of the comb filter is based on the frequency interleaving shown in Fig.4.1 . As shown in Fig.5.1, the luminance comb filter suppresses odd harmonics of horizontal scan rate and the chrominance comb filter suppresses even harmonics of the horizontal scan rate. The comb filter is available in several variants, offering different quality for different computational complexities. Some of them are discussed below

Two Line Comb Filter

Simplest of all variants, the action of this comb filter is based on the assumption that if color content is more or less the same on successive lines in a field, there will be a phase difference of π radians in the chrominance values of the corresponding pixels. Hence if we add two successive scan lines, the resulting sum will have the chrominance information cancelled. To see this effect mathematically, consider a pixel with chrominance amplitude X and chrominance angle *theta*. When this pixel is delayed by one scan duration $\frac{1}{f_H}$ and added to the corresponding pixel in the next scan line, the resultant can be written as

$$\begin{aligned} C[n] + C[n-1] &= Xcos(2\pi f_{sc}t + \theta) + Xcos(2\pi f_{sc}(t - \frac{1}{f_H}) + \theta) \\ &= Xcos(2\pi f_{sc}t + \theta) + Xcos(2\pi f_{sc}t + \theta - \frac{455\pi f_H}{f_H}) \\ &= 0 \end{aligned} \tag{5.2}$$

However if the chrominance information changes on successive scan lines, there will be imperfect chroma cancellation, which results in an effect know as dot crawl.

Three Line Comb Filter

A three line comb filter uses three successive lines instead of two for luma/chroma separation. It results in reduced dot crawl but also results in a loss of vertical resolution.

Adaptive Two Line Comb Filter

The adaptive two line comb filter structure shown in Fig. xx computes the correlation between the current line and the delayed line and between the current line and the next line and decides which pair of lines to use for Y/C separation. In the three variants described here, adaptive version offers the best performance albeit at a higher computational cost.

In addition to the comb filters described here, motion adaptive comb filters are also in use in the industry. These comb filters offer almost perfect Y/C separation but again at a high computational cost.

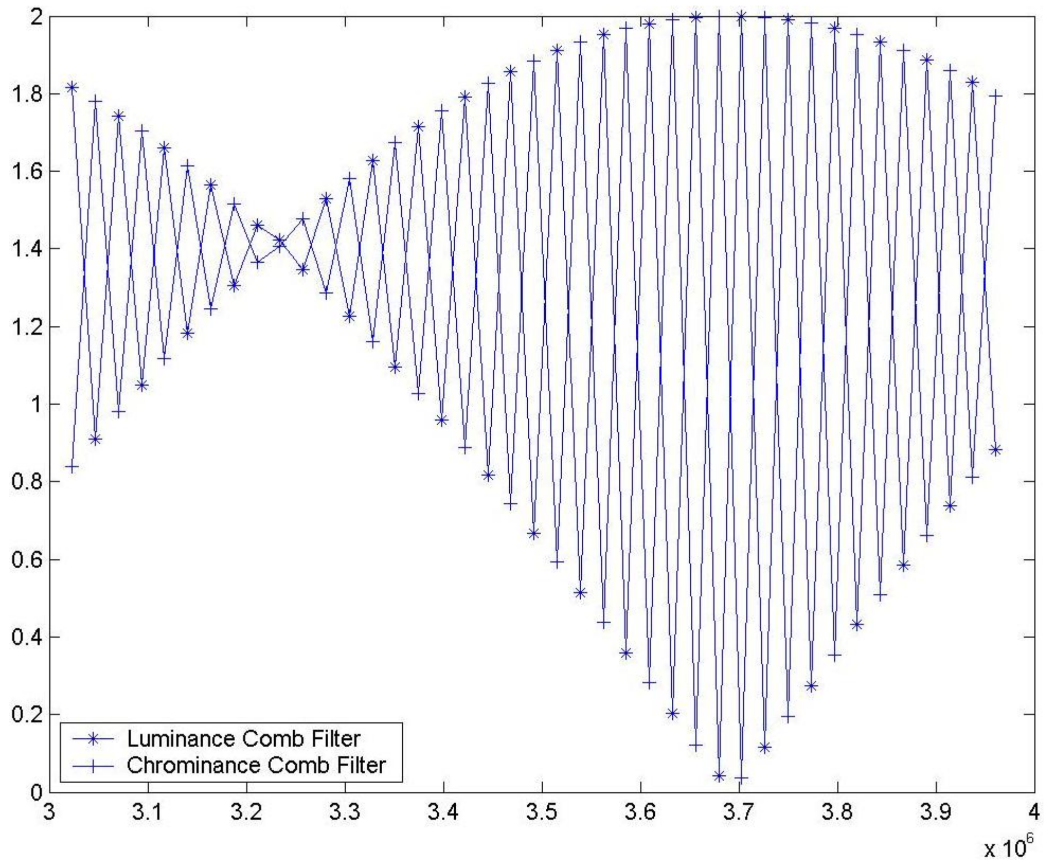


Figure 5.1: Comb Filter Action

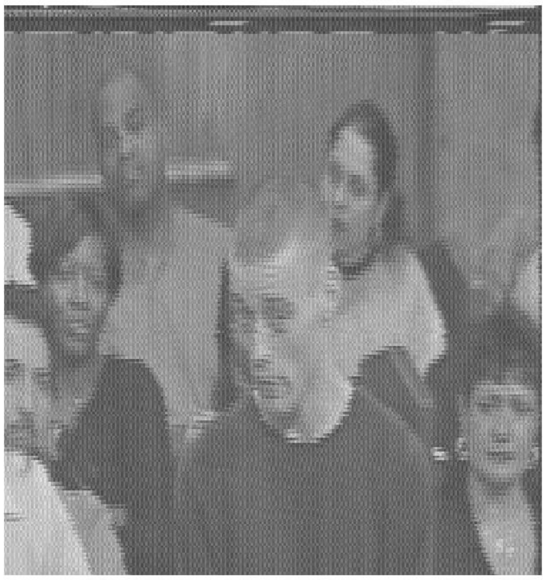


Figure 5.2: Aliasing Effect

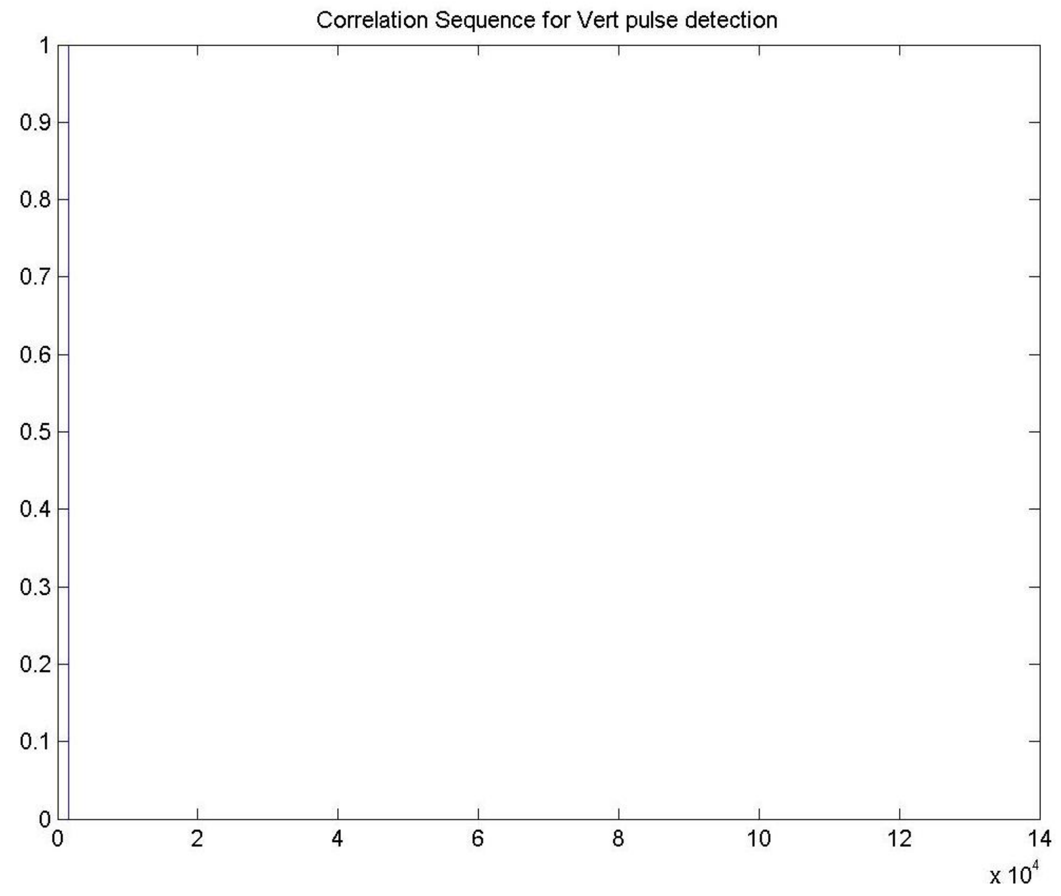


Figure 5.3: Correlation sequence for first V-Sync identification

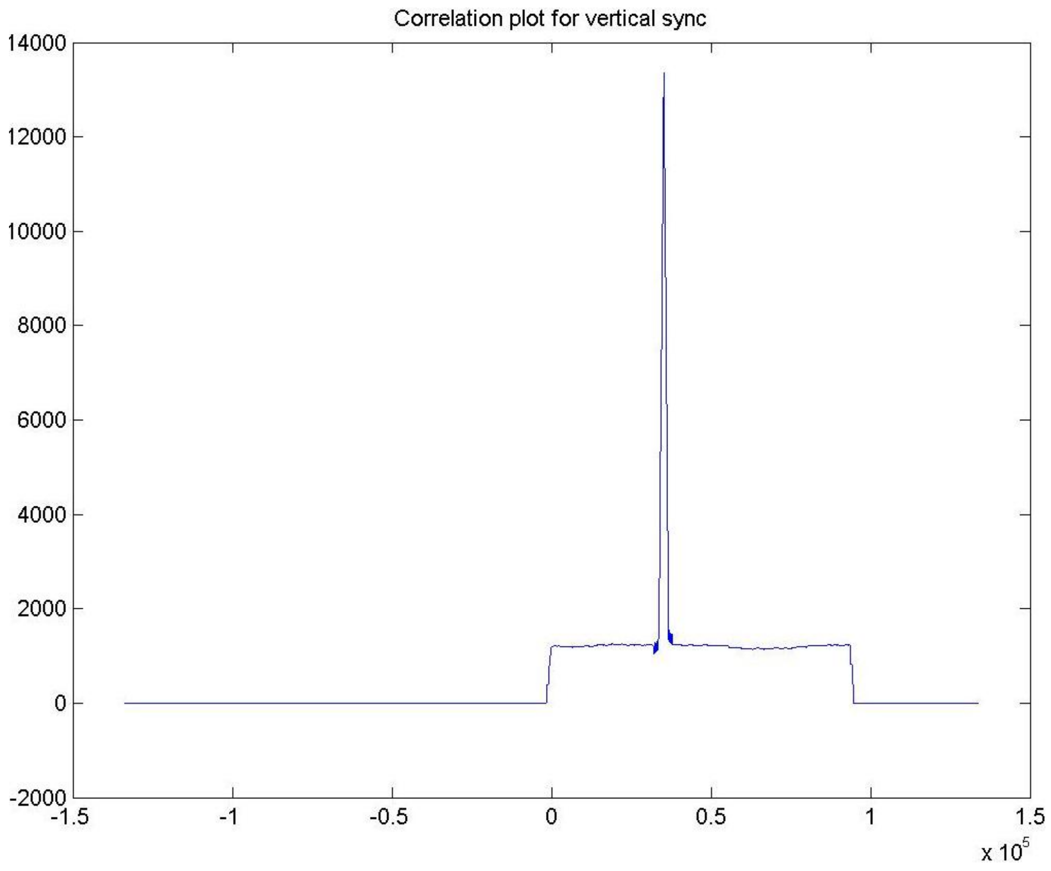


Figure 5.4: Correlation result for V-Sync Detection

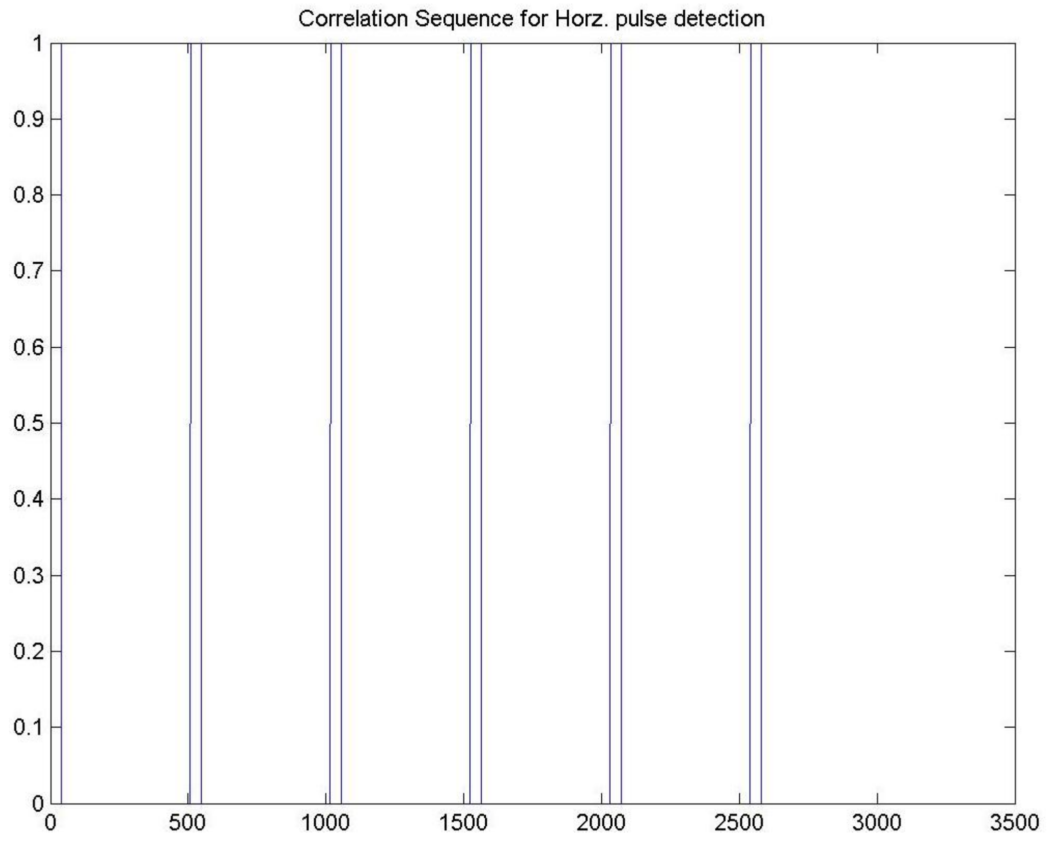


Figure 5.5: Correlation sequence for first H-Sync identification

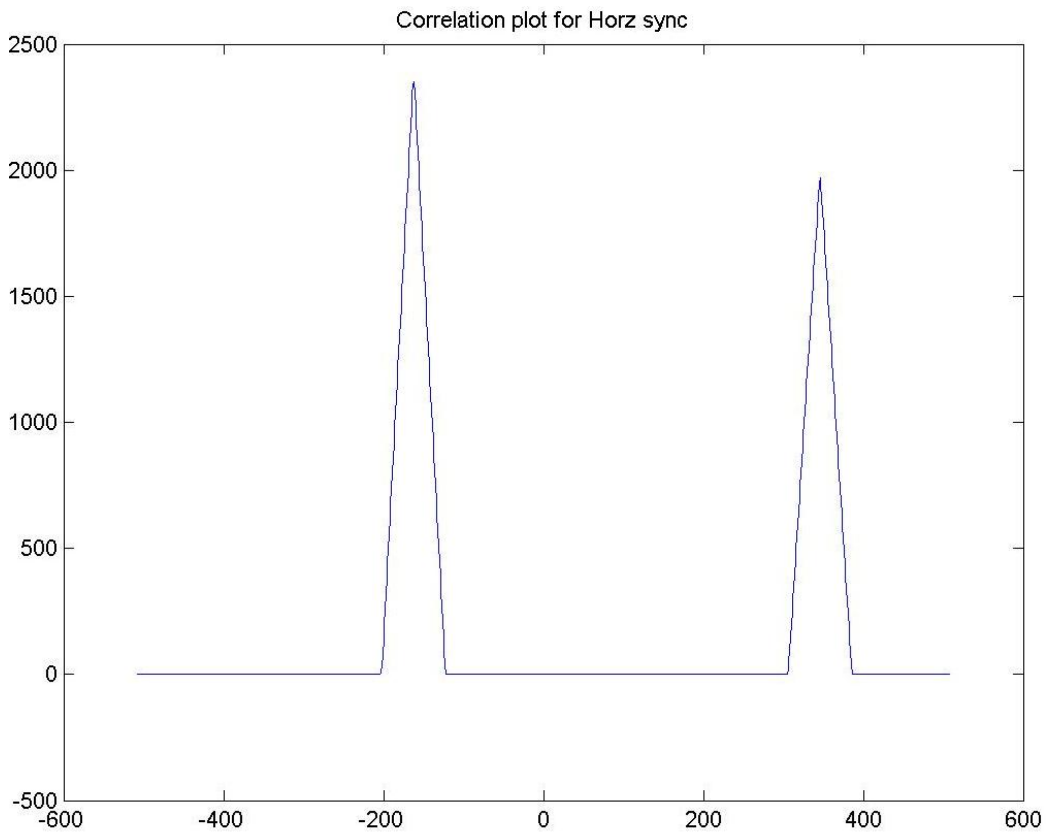


Figure 5.6: Correlation result for H-Sync Detection

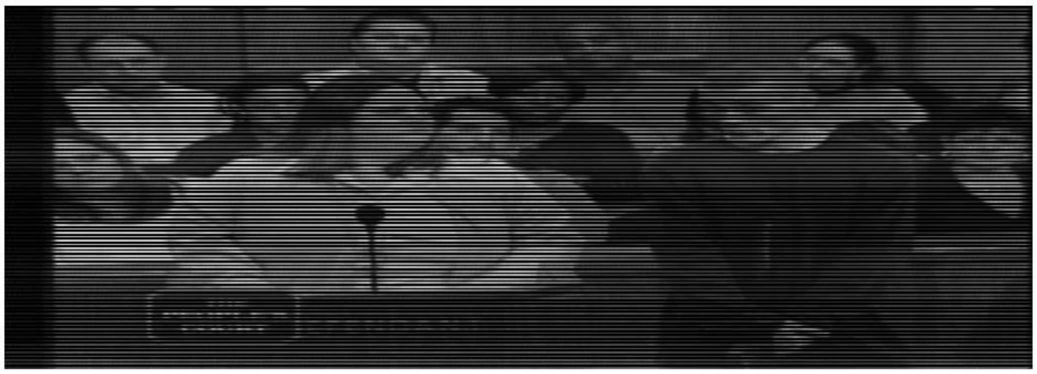


Figure 5.7: Even Field

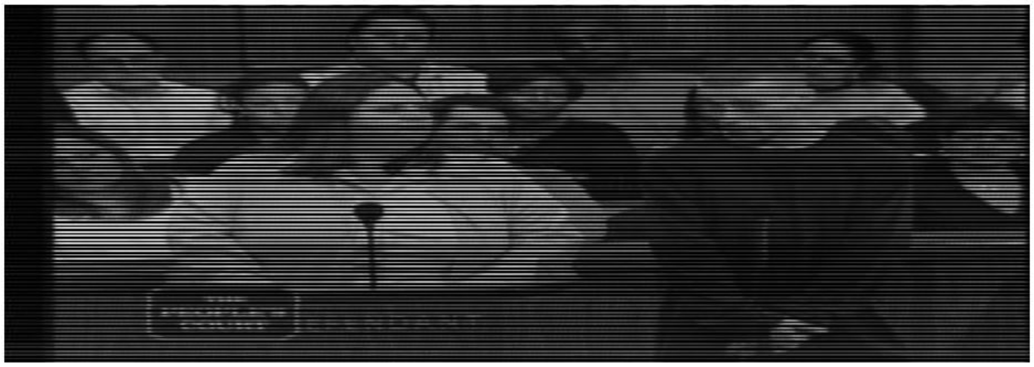


Figure 5.8: Odd Field

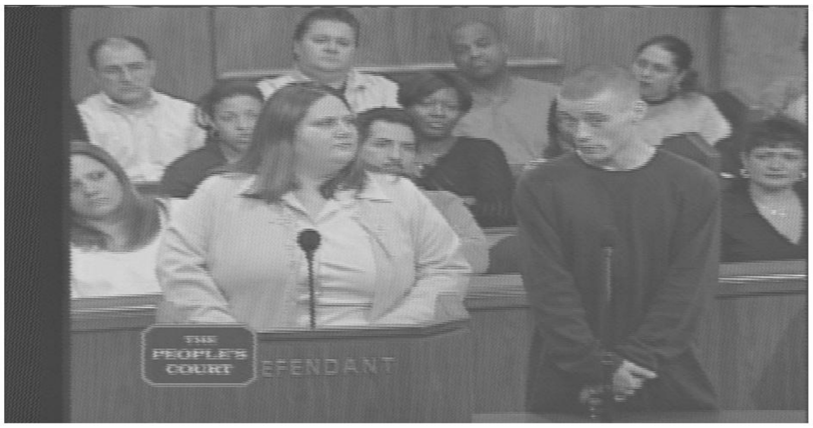


Figure 5.9: De-interleaved monochrome image before applying Comb Filter

Chapter 6

Color Decoding

This chapter discusses theoretical aspects of color decoding.

6.1 Chrominance Comb Filter

The chrominance comb filter is obtained by subtracting the delayed pixel instead of adding it in as in (5.2). This has the effect of cancelling out the luminance component and preserving the chrominance component. Again the method is as accurate as the timing information and color subcarrier frequency.

6.2 Color Burst

Color burst is located on the back porch of the blanking interval and is 8 to 11 cycles of unmodulated carrier, present to provide an amplitude and phase reference for DSBSC synchronous decoding. Peak to Peak amplitude of color subcarrier is equal to horizontal sync amplitude.

6.3 YIQ Modulation

Three primary colors Red, Green and Blue are required to construct a color frame. However, to maintain backward compatibility, NTSC signal uses luminance (Y) and two other signals, inphase (I) and quadrature (Q) signals, derived from RGB and Y. The YIQ signals are derived from RGB by using the following formula

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.30 + 0.59 + 0.11 \\ 0.60 - 0.28 - 0.32 \\ 0.21 - 0.51 + 0.30 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{6.1}$$

The inverse transformation can be used to convert from YIQ to RGB space. The I and Q signals are modulated on the color subcarrier using the following expression

$$C(t) = I(t)cos(2\pi f_{sc}t + 33^o) + Q(t)sin(2\pi f_{sc}t + 33^o) \tag{6.2}$$

Therefore I/Q information is encoded into the phase and amplitude of the DSBSC carrier. Fig.6.1 shows the phase of the chrominance component of a single scan line. As expected, the phase for first 50 samples, corresponding to the color burst varies linearly with time, with a slope equal to $2\pi f_{sc}$.

6.4 DSBSC Demodulation

To demodulate the DSBSC modulated chrominance information, receiver uses phase reference from the color burst to subtracts it from the phase of sampled pixels, simultaneously compensating for the factor $2\pi f_{sc}t$. Using this compensated phase information and amplitude information, I and Q components are estimated and linear transformation described in (6.1) is carried to get back the original RGB co-ordinates.

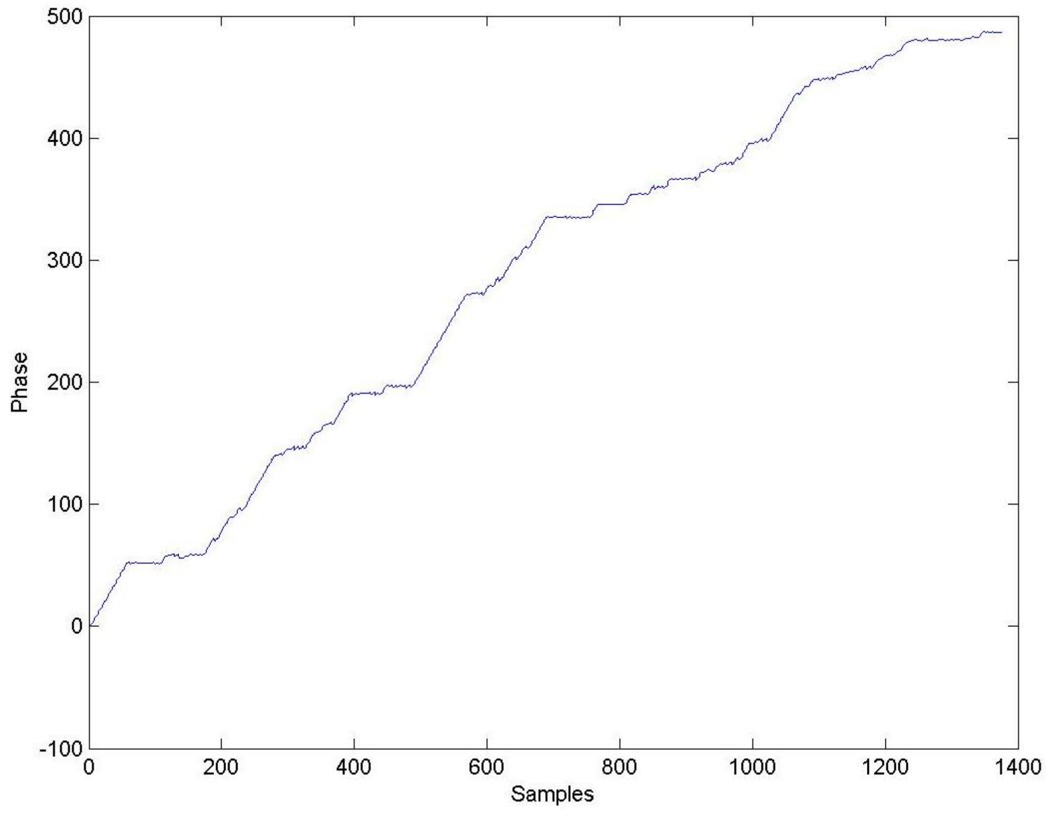


Figure 6.1: Phase plot of a chrominance image row

Chapter 7

Experimental Results and Discussion

This chapter discusses the experimental results obtained and discusses several implementation issues that were faced during the course of this work.

7.1 Comb Filter Results

Fig.7.1 - Fig.7.3 shows the output of various luminance comb filter variants discussed earlier in the thesis. Fig.7.4 shows the spectrum of the 2 line luminance comb filter's output image. Fig.7.5 shows the output image of the the 2 line chrominance comb filter and Fig.7.6 shows the spectrum of this image.

Comb filter performance is highly dependent on accurate timing information and frequency estimation accuracy. Fig.7.9 shows the degraded output image of the 2 line luminance comb filter when frequency of the color subcarrier is slightly shifted from the standard 3.58 MHz value. Most TVs switch to bandpass filtering mode when the source frequency is not very stable, as the case is with some VCRs.

7.2 Color Decoding Results

Fig.7.7 shows the decoded color frame and Fig.7.8 on page 49 shows the decoded frame when the factor $2\pi f_{sc}t$ is not properly compensated for.

7.3 Implementation Issues

This section discusses several implementation issues that need to be addressed when implementing the proposed color decoding system.

7.3.1 FIR v/s IIR Filtering

Filtering is required in the receiver to suppress the audio carrier and also when decimating or interpolating the data streams. Although IIR filters are computationally much more efficient than FIR filters, they lead to envelope distortion due to their non-linear phase response. Hence IIR filtering is not advisable in situations where the information is available as the signal envelope. Also when using FIR filters, different streams may have different group delays based on filter length. It is therefore important to account for these delays when working on such streams together. For example in the proposed receiver structure different streams are used for extracting the timing information and for drawing the frame. When using both these informations together, synchronization must be properly maintained.

7.3.2 Picture Carrier Frequency Drift

As explained mathematically in the section on quadrature sampling, any difference in picture carrier and local oscillator frequency and phase will manifest itself in the samples as resultant frequency $f_c - f_{LO}$ and phase θ. In order to achieve good Y/C separation, these differences should be minimized. In the proposed receiver structure this drift has been compensated for based on an estimate of the average frequency drift over one frame.

Apart from frequency and phase relationship between the local oscillator and the picture carrier, color decoding performance is also sensitive to the drift of the color subcarrier. A 1 Hz drift or estimation error in the color subcarrier frequency can lead to pixel color traversing the entire rainbow in 1 second. A 60 Hz estimation will produce discernible rainbow bands all across the image. Therefore very tight control has to be maintained on frequency of the local oscillator and precise estimate of the color subcarrier drift is needed for vivid and faithful color reproduction.

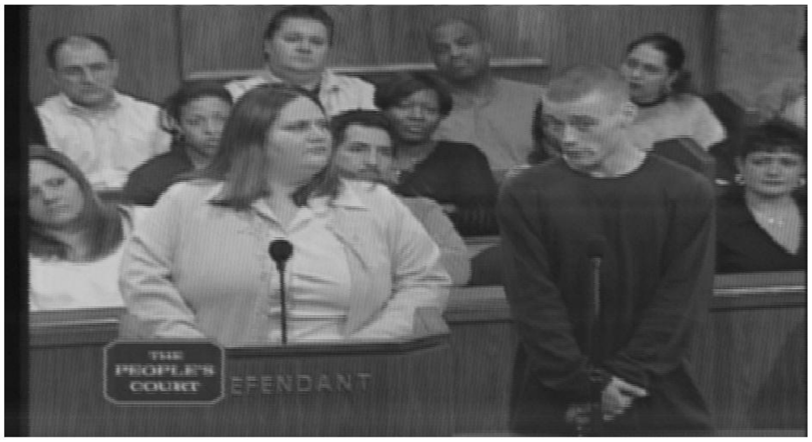


Figure 7.1: Luminance Image from a 2 line comb filter

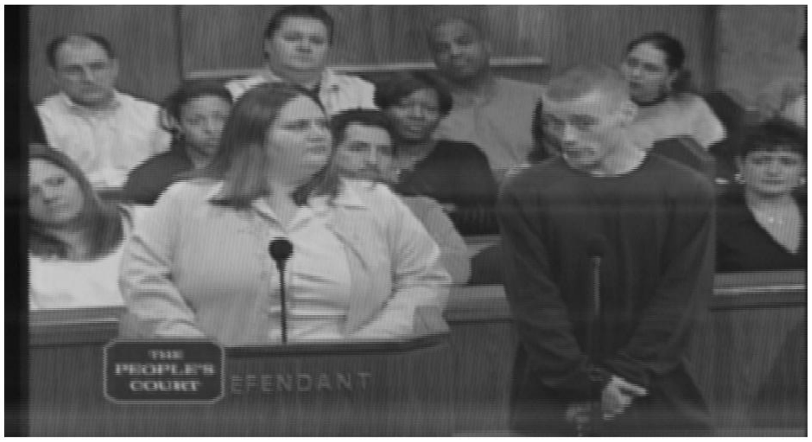


Figure 7.2: Luminance Image from a 3 line comb filter

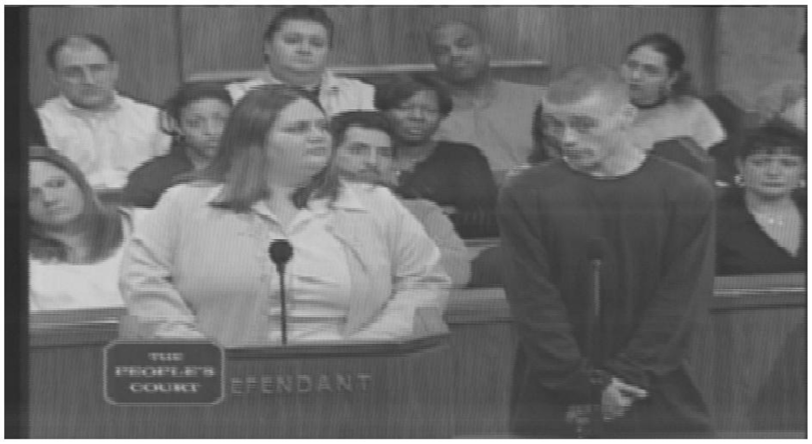


Figure 7.3: Luminance Image from an adaptive 2 line comb filter

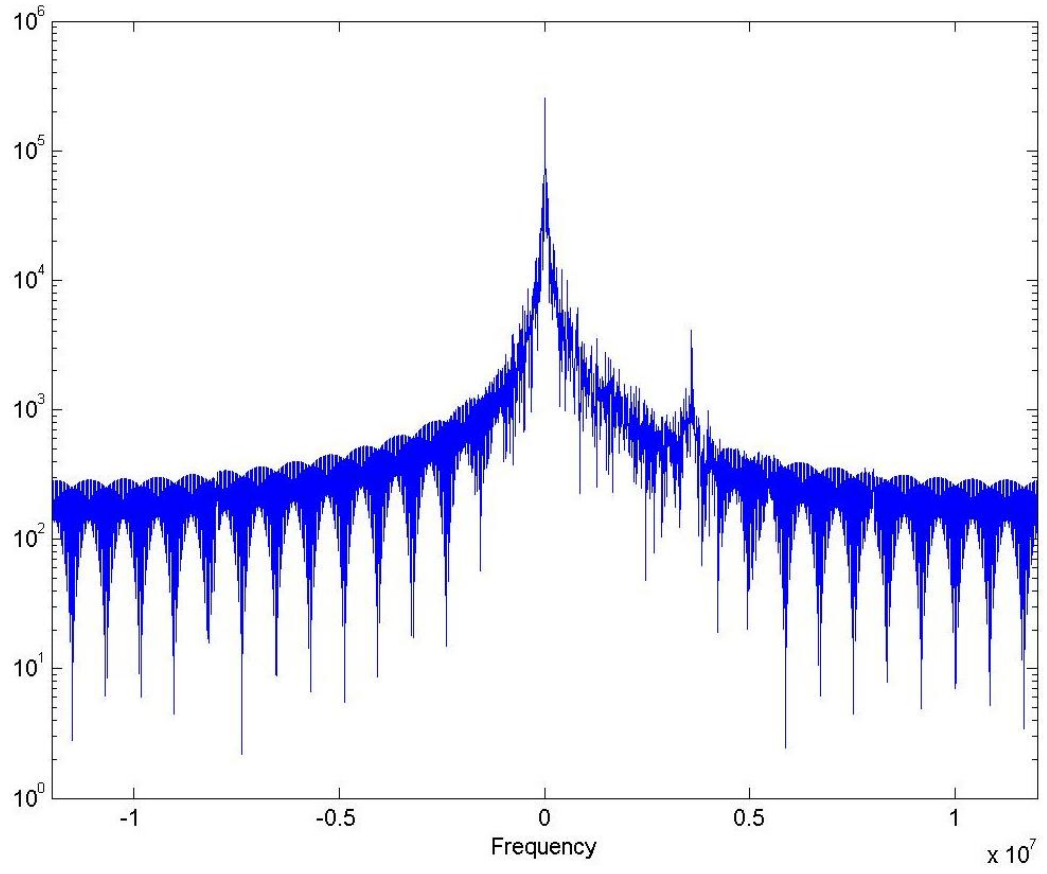


Figure 7.4: Luminance Image FFT

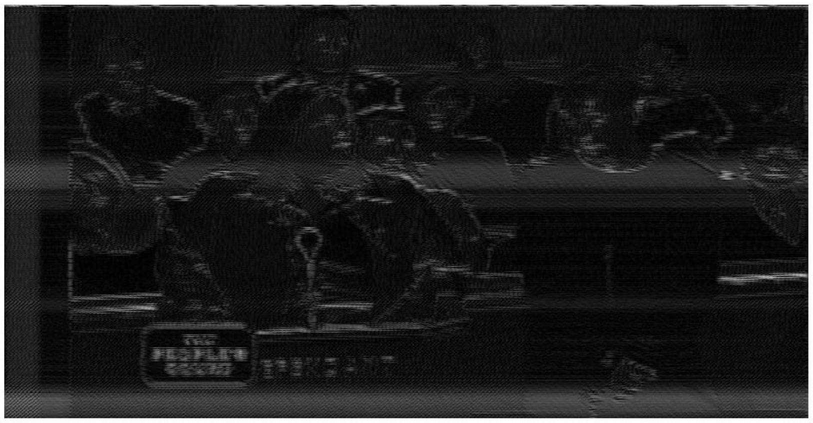


Figure 7.5: Chrominance Image from 2 line comb filter

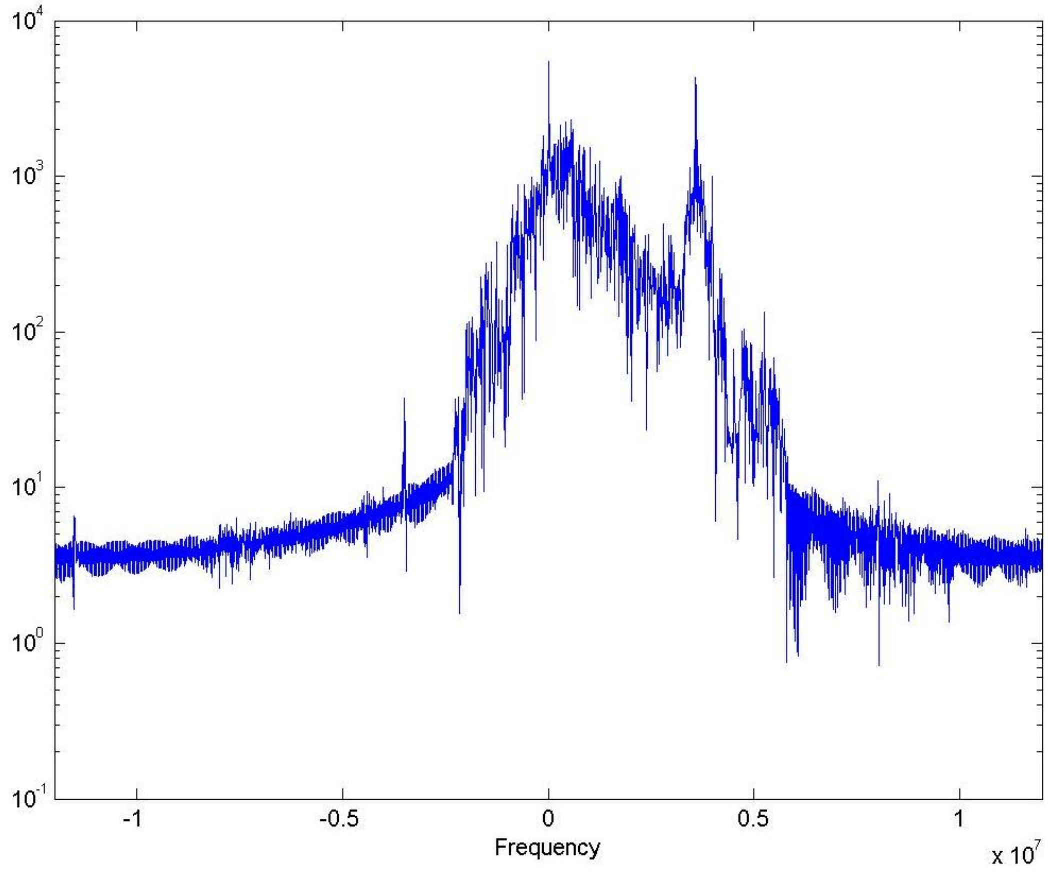


Figure 7.6: Chrominance Image FFT

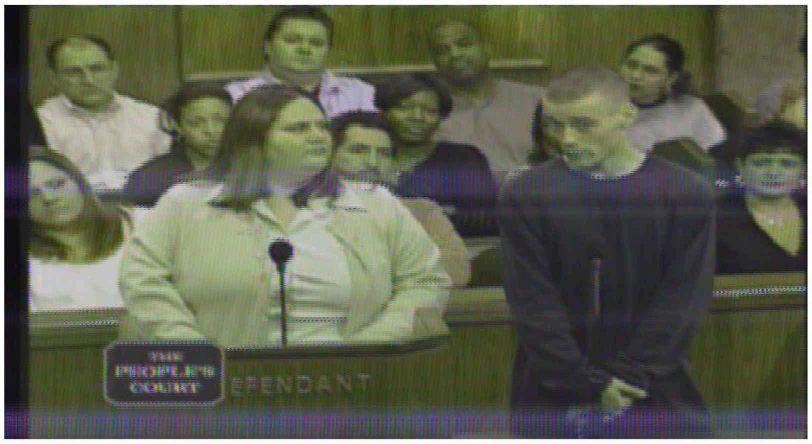


Figure 7.7: Full Color Frame

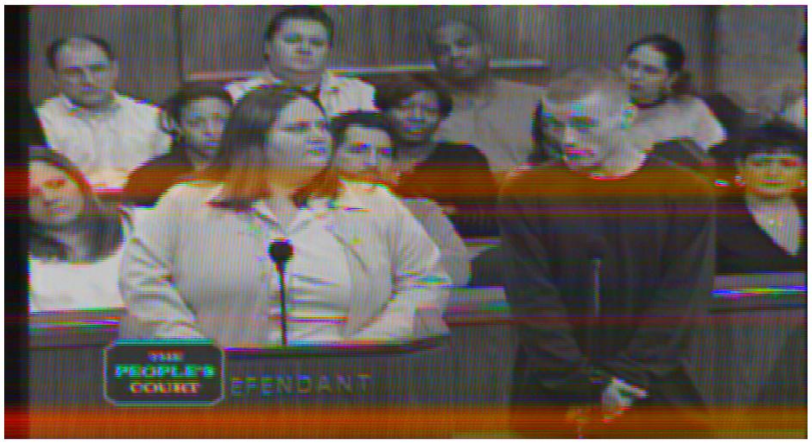


Figure 7.8: Improperly Phase Compensated Color Frame

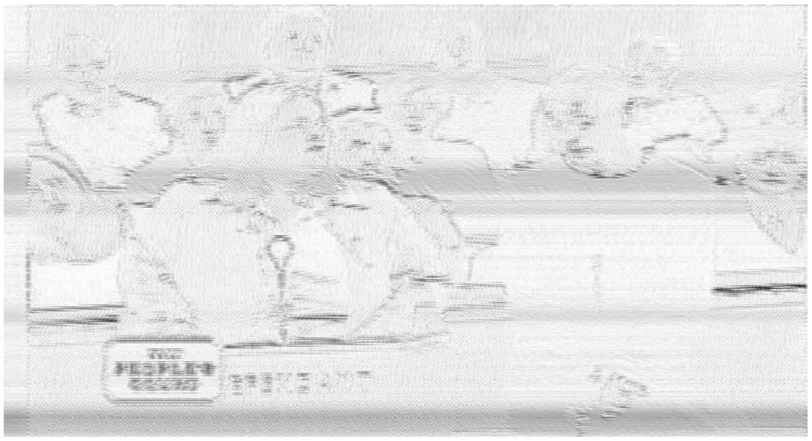


Figure 7.9: Degraded Comb Filter Output Due to Frequency Drift

Chapter 8

Conclusions

The two semester long work focused on studying GNU Radio as a platform for software radio application development and developing an application for decoding NTSC video and audio. In the first phase of the project, GNU Radio was installed on a Linux based system and NTSC audio decoding was successfully tried using it. In the second phase of the project, NTSC monochrome and color decoding algorithms were developed and implemented in Matlab and were successfully tried on offline NTSC samples. The GNU Radio code for NTSC video decoding is currently under development. A monochrome video comprising of about 116 frames along with the audio was put up as an Audio Video Interleaved (AVI) file and is available at [11].

Chapter 9

Directions for Future Work

Paucity of time prevented trying a lot of things which could have given great results. Certain areas where more effort can be put in are mentioned below

- The algorithm needs to be tested on SMPTE color bars and fine tuned accordingly.
- A Digital PLL should be implemented to compensate for the frequency drift of the picture/color subcarrier. This should drastically improve both monochrome and color decoding performance.
- Motion adaptive comb filters can be implemented to give excellent color decoding performance.
- Work can be done on reducing the computational complexity of the algorithm with the objective of making the algorithm amenable to real time implementation.

Bibliography

[1] "Vanu, Inc.Homepage," Available at: www.vanu.com.

[2] "GNU Radio Project Homepage," Available at: www.gnu.org/software/gnuradio/ .

[3] M. Ettus, "Ettus Research LLC Homepage," Available at www.ettus.com.

[4] R. Lyons, *Understanding Digital Signal Processing.* Pearson Education, Second Edition, 2004.

[5] "Microtune RF Tuner Modules: 49xx Series Homepage," Available at www.microtune.com/products/49xx_Series.html.

[6] S. Haykin, *Communication Systems.* John Wiley and Sons, Inc., Fouth Edition, 2001.

[7] "Python Programming Language Homepage," Available at www.python.org.

[8] "SWIG (Simplified Wrapper and Interface Generator) Homepage," Available at www.swig.org.

[9] R. R. Gulati, *Monochrome and Colour Television.* New Age International (P) Limited Publishers, Reprint, 2004.

[10] A. Luther and A. Inglis, *Video Engineering*. McGraw Hill Publishers, Third Edition, 1999.

[11] P. Dayal, “NTSC Decoding Status Page,” Available at www.geocities.com/pmd_iitgw/software_radio.htm.