THE RMS PHASE ERROR OF A PHASE-LOCKED LOOP
FM DEMODULATOR FOR STANDARD NTSC VIDEO

by

DALE F. DUBBERT
B.S., Kansas State University, 1984

A MASTER'S REPORT

submitted in partial fulfillment of
the requirements for the degree

MASTER OF SCIENCE

Department of Electrical and Computer Engineering
KANSAS STATE UNIVERSITY
Manhattan, Kansas

1986

Approved by:

[Signature]
Major Professor
Table of Contents

List of Figures.................................................. iii
List of Tables.................................................. v
Acknowledgements.............................................. vi

I. Introduction.................................................. 1

II. System Overview............................................. 3

III. Computer Simulation of RMS Phase Error................. 5
   NTSC Color Bar Generator.................................. 5
   FM Preemphasis............................................. 11
   Integration.................................................. 13
   Phase-Locked Loop Error Function......................... 17
   RMS Phase Error Calculation............................... 20
   Signal Analysis Routines................................... 21

IV. Experimental Measurement of Phase Error................. 26

V. Comparison of Measured and Simulated System Performance... 30

VI. Phase Error Results........................................ 40

VII. Satellite FM Receiver..................................... 45

VIII. Conclusions and Future Suggested Studies................ 47

IX. Bibliography................................................ 48

Appendix 1. Software Listing.................................... 49
Appendix 2. Program Operating Instructions.................... 98
Appendix 3. Hardware Schematics................................ 107
THIS BOOK CONTAINS SEVERAL DOCUMENTS THAT ARE OF POOR QUALITY DUE TO BEING A PHOTOCOPY OF A PHOTO.

THIS IS AS RECEIVED FROM CUSTOMER.
THIS BOOK CONTAINS NUMEROUS PAGES WITH DIAGRAMS THAT ARE CROOKED COMPARED TO THE REST OF THE INFORMATION ON THE PAGE. THIS IS AS RECEIVED FROM CUSTOMER.
THIS BOOK CONTAINS NUMEROUS PAGES WITH THE ORIGINAL PRINTING BEING SKEWED DIFFERENTLY FROM THE TOP OF THE PAGE TO THE BOTTOM.

THIS IS AS RECEIVED FROM THE CUSTOMER.
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Block diagram of the phase error measurement system</td>
<td>4</td>
</tr>
<tr>
<td>3.1</td>
<td>NTCS encoder</td>
<td>6</td>
</tr>
<tr>
<td>3.2</td>
<td>NTSC encoder inputs for NTSC standard 100% saturated color bars (single horizontal scan)</td>
<td>7</td>
</tr>
<tr>
<td>3.3</td>
<td>Equivalent low pass model of the NTSC encoder</td>
<td>10</td>
</tr>
<tr>
<td>3.4</td>
<td>Single horizontal scan of the unfiltered composite NTSC color bar pattern</td>
<td>12</td>
</tr>
<tr>
<td>3.5</td>
<td>Gain characteristics for the CCIR 405-1 FM preemphasis and deemphasis filters for a 525 line TVRO system</td>
<td>14</td>
</tr>
<tr>
<td>3.6</td>
<td>Phase characteristics for the CCIR 405-1 FM preemphasis and deemphasis filters for a 525 line TVRO system</td>
<td>15</td>
</tr>
<tr>
<td>3.7</td>
<td>Equivalent low pass linear model of the FM transmitter (preemphasis plus direct FM modulation) and the phase-locked loop error model</td>
<td>16</td>
</tr>
<tr>
<td>3.8</td>
<td>Phase-locked loop linear model for small phase error</td>
<td>18</td>
</tr>
<tr>
<td>3.9</td>
<td>Method of obtaining the time domain PLL phase input ( \theta(t) ) and the magnitude of the FM spectra (</td>
<td>\hat{s}(\omega)</td>
</tr>
<tr>
<td>3.10</td>
<td>Equivalent low pass model for obtaining the PLL demodulated signal</td>
<td>25</td>
</tr>
<tr>
<td>4.1</td>
<td>The four major components employed in the experimental phase error measurements</td>
<td>29</td>
</tr>
<tr>
<td>4.2</td>
<td>Lab set up for the experimental phase error measurements</td>
<td>29</td>
</tr>
<tr>
<td>5.1</td>
<td>Measured time domain video signal with preemphasis (top) and without preemphasis (bottom)</td>
<td>31</td>
</tr>
<tr>
<td>5.2</td>
<td>Simulated NTSC color bar test signal with chrominance and luminance low pass filters applied</td>
<td>32</td>
</tr>
<tr>
<td>5.3</td>
<td>Simulated NTSC test signal with FM preemphasis</td>
<td>33</td>
</tr>
<tr>
<td>5.4</td>
<td>Measured video spectrum for the preemphasized NTSC color bar pattern</td>
<td>34</td>
</tr>
<tr>
<td>5.5</td>
<td>Simulated spectrum of the preemphasized NTSC test signal</td>
<td>35</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>5.6</td>
<td>Measured FM spectrum for the NTSC test signal with Δf = 10.7 MHz</td>
<td>36</td>
</tr>
<tr>
<td>5.7</td>
<td>Simulated FM spectrum for the NTSC test signal with Δf = 10.7 MHz</td>
<td>37</td>
</tr>
<tr>
<td>5.8</td>
<td>Simulated time domain PLL phase input</td>
<td>38</td>
</tr>
<tr>
<td>5.9</td>
<td>Example of a demodulated test signal with deemphasis applied</td>
<td>39</td>
</tr>
<tr>
<td>6.1</td>
<td>Theoretical PLL demodulator frequency response curves for the nine tested values of $f_n$</td>
<td>42</td>
</tr>
<tr>
<td>6.2</td>
<td>RMS phase error data for the software simulation and the hardware tests</td>
<td>43</td>
</tr>
<tr>
<td>7.1</td>
<td>Satellite television receiver employing a second order PLL video demodulator</td>
<td>46</td>
</tr>
</tbody>
</table>
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Specifications for the NTSC encoder low pass filters</td>
<td>9</td>
</tr>
<tr>
<td>6.1</td>
<td>Summary of the fixed system parameters for the experimental and simulated phase error measurements</td>
<td>41</td>
</tr>
</tbody>
</table>
Acknowledgements

The author would like to express his sincere thanks to Dr. Gail Simons, Dr. Donald Lenhert, and Dr. Donald Hummels for their participation as graduate committee members. Additional thanks are extended to Dr. Hummels for his duties as major professor and his encouragement and patience throughout this project. I would also like to extend a special thanks to Dr. Corwin Bennett for his guidance and encouragement throughout my undergraduate as well as graduate work.

The efforts of Brenda Merryman for typing the manuscript and Gina Pacumbaba for drafting the hardware schematics are also greatly appreciated.
I. Introduction

The RMS phase error of a phase-locked loop (PLL) is a major consideration when designing a PLL FM demodulator. The general rule is to keep the phase error small thus allowing the PLL to maintain a linear phase error transfer relationship. This both simplifies the theoretical analysis of the PLL and prevents distortion due to phase detector nonlinearity and loss of lock.

The loop bandwidth must be selected to keep the phase error small (wide loop bandwidth) and at the same time keeping the noise bandwidth narrow enough to result in an acceptable output signal to noise ratio (narrow loop bandwidth). In the context of this report, the undamped natural frequency $f_n$ of the PLL linear model is considered as the independent parameter and the major factor in determining the demodulator loop bandwidth. Both computer analysis and experimental work are employed to study the optimum range of $f_n$ when designing a PLL FM demodulator. Since, in this report, quantitative signal-to-noise-ratio and threshold measurements are not evaluated, the optimum value of $f_n$ is considered to be the minimum frequency for which the phase error remains relatively small. Below this value, the RMS phase error $E_{\text{RMS}}$ increases drastically for decreasing $f_n$.

As mentioned, both software and hardware measurements were made and compared thus verifying the accuracy of each. The test signal in each case was chosen to be a standard NTSC (National Television Systems Committee) 100% saturated color bar test pattern. The software simulation employs linear, frequency domain filtering techniques and equivalent low pass modeling to process the chrominance (color) information. The luminance (monochrome) information is processed at
baseband. Separation of the chrominance and luminance signals allows a more efficient means of processing the data and allows the programmer, if so desired, to study the RMS phase error effects of the chrominance and luminance signals separately. The hardware implements the software simulation by employing a PLL designed and built for use in a satellite FM receiver. The NTSC color bar generator, FM pre-emphasis filter and the 70 MHz FM modulator were designed and built solely for the purpose of phase error testing.
II. System Overview

The phase-locked loop RMS phase error was determined by experimental testing and computer analysis using the system illustrated in Figure 2.1. Both hardware tests and software simulations were employed to test the PLL phase error for a fixed NTSC test pattern. In both cases, noise, 70 MHz IF prefiling and audio subcarriers are not considered as part of the system. The addition of audio subcarrier(s) would present additional system parameters such as the audio subcarrier frequency, the message content and, most importantly, the relative amplitude of the subcarrier.

The major objective was to acquire RMS phase error data for the PLL demodulator as a function of the undamped natural frequency $f_n$ of the PLL linear model. The natural frequency was chosen as the independent parameter due to its large effect on the loop phase error. The remaining system parameter values are based either on fixed hardware parameters (PLL open loop gain, FM transmitter conversion gain, etc.) or standards set by the NTSC or the CCIR (International Radio Consultative Committee).
Figure 2.1. Block diagram of the phase error measurement system.
III. Computer Simulation of RMS Phase Error

Referring again to Figure 2.1, the computer model consists of four major components: 1) the NTSC color bar generator, 2) FM preemphasis, 3) the FM modulator (VCO), and 4) the PLL video demodulator. A simulation of the model was written in FORTRAN and run on a VAX 11/750. The simulation employs primarily frequency domain linear filtering techniques as well as equivalent low pass modeling.

NTSC Color Bar Generator

The block diagram of the NTSC standard for encoding color television signals is shown in Figure 3.1. For the case of an NTSC standard 100% saturated color bar pattern, the red, green and blue camera inputs \((r(t), g(t), b(t))\) and \((b(t))\) respectively) consist of 1 volt peak-to-peak square wave pulses as shown in Figure 3.2. The horizontal synchronization signal \(p(t)\) consists of a horizontal sync pulse plus the horizontal blanking pedestal. The color burst signal is formed by modulating the color burst pulse \(u(t)\) on the color subcarrier frequency \(f_c\). The NTSC specification for \(f_c\) is

\[
f_c = 3.579545 \text{ MHz}.
\]

The color subcarrier frequency is exactly 227.5 times the horizontal line frequency \(f_h\) which is

\[
f_h = 15.73426 \text{ kHz}.
\]

Referring again to Figure 3.1, the linear transformation matrix \(A\) is used to form the I and Q chrominance signals \((i(t)\) and \(q(t)\) respectively) and the Y luminance signal \(y(t)\). The linear transformation is
Figure 3.1 NTSC encoder.
Figure 3.2 NTSC encoder inputs for NTSC standard 100% saturated color bars (single horizontal scan).
expressed as

\[
\begin{bmatrix}
1(t) \\
q(t) \\
y(t)
\end{bmatrix} =
\begin{bmatrix}
0.600 & -0.377 & -0.221 \\
0.191 & -0.558 & 0.368 \\
0.299 & 0.587 & 0.114
\end{bmatrix}
\begin{bmatrix}
r(t) \\
g(t) \\
b(t)
\end{bmatrix}.
\] (3.1)

The chrominance signals \(i(t)\) and \(q(t)\) are then low pass filtered and modulated in phase quadrature. The color burst signal is then added to form the bandpass chrominance signal \(c(t)\).

The luminance signal consists of the sum of the \(Y\) signal \(y(t)\) and the horizontal synchronization signal \(p(t)\), i.e.,

\[
l(t) = y(t) + p(t).
\] (3.2)

Both \(y(t)\) and \(p(t)\) are low pass filtered prior to forming the luminance signal \(l(t)\).

To facilitate a more efficient means of processing the chrominance signal \(c(t)\), it was decided to express the signal in terms of its low pass equivalent \(\hat{c}(t)\) given as

\[
\hat{c}(t) = \left[2z(t) e^{-j\omega_c t}\right]_{\text{LPT}},
\] (3.3)

where \(\omega_c = 2\pi f_c\)

LPT indicates retention of only the low pass term.

From inspection of Figure 3.1, we can write

\[
c(t) = \frac{1}{2} \left\{ i_f(t) \cos \omega_c t + q_f(t) \sin \omega_c t \right\},
\] (3.4)

where \(i_f(t)\) and \(q_f(t)\) denote the low pass filtered versions of \(i(t)\) and \(q(t)\) respectively.

As previously mentioned, the color burst pulse \(u(t)\) is also modulated on the color subcarrier for use as a color phase reference. The phase of the color burst signal is 180° with respect to \(\sin \omega_c t\). By combini-
ing the color burst with the existing chrominance signal, equation 3.4 can be rewritten as

\[ c(t) = \frac{1}{2} \left\{ i_f(t) \cos \omega_c t + \left[ q_f(t) - u(t) \right] \sin \omega_c t \right\}. \quad (3.5) \]

The equivalent low pass version of equation 3.5 is obtained by applying equation 3.3 and simplifying:

\[ \tilde{c}(t) = \frac{1}{2} \left\{ i_f(t) - j[q_f(t) - u(t)] \right\}. \quad (3.6) \]

Figure 3.3 illustrates the equivalent low pass model of the NTSC encoder. Both \( u(t) \) and \( l(t) \) are filtered by a 3-pole Butterworth low pass video filter denoted by \( H_v(\omega) \). The equivalent low pass chrominance signal \( \tilde{c}(t) \) is filtered by the frequency shifted version of the video filter denoted by \( H_v(\omega + \omega_c) \). Specifications for the chrominance and luminance filters are summarized in Table 3.1.

**Table 3.1 Specifications for the NTSC encoder low pass filters.**

<table>
<thead>
<tr>
<th>Filter</th>
<th>Filter Type</th>
<th>Number of Poles</th>
<th>3 dB Cutoff Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>( H_v(\omega) )</td>
<td>Butterworth low pass</td>
<td>3</td>
<td>4.2 MHz</td>
</tr>
<tr>
<td>( H_i(\omega) )</td>
<td>Butterworth low pass</td>
<td>3</td>
<td>1.5 MHz</td>
</tr>
<tr>
<td>( H_q(\omega) )</td>
<td>Butterworth low pass</td>
<td>3</td>
<td>0.5 MHz</td>
</tr>
</tbody>
</table>

The outputs of the NTSC encoder are the baseband luminance signal \( l(t) \) defined by equation 3.2 and the equivalent low pass chrominance signal \( \tilde{c}(t) \) defined by equation 3.6. The composite video signal \( v(t) \) can be calculated from \( l(t) \) and \( \tilde{c}(t) \) by the following equation:

\[ v(t) = l(t) + |\tilde{c}(t)| \cos[\omega_c t + \angle \tilde{c}(t)], \quad (3.7) \]
Figure 3.3. Equivalent low pass model of the NTSC encoder.
where \( \phi \) denotes phase angle in radians.

Figure 3.4 illustrates the composite signal \( v(t) \) for the case where both the luminance and the chrominance filters are inactive. The figure illustrates a single horizontal scan (63.56 \( \mu \text{sec} \)) which is periodic with a fundamental frequency of \( f_h \).

**FM Preemphasis**

To comply with current standards for satellite FM transmission, the chrominance and luminance outputs of the NTSC encoder must be preemphasized to improve the noise performance of the FM system. The preemphasis circuit shown in Appendix 3 is as specified by the International Radio Consultative Committee (CCIR) recommendation 405-1 for a 525 line TVRO (television receive only) system. Appendix 3 also gives the component values for the FM deemphasis network employed in the TVRO demodulator.

By applying nodal analysis to the preemphasis \( H_p(\omega) \) or deemphasis \( H_d(\omega) \) network, a general transfer function \( H(\omega) \) can be written as

\[
H(\omega) = \frac{1}{4} \frac{(2 + \frac{R}{Z2})(\frac{R}{Z1}) + 1}{(1 + \frac{R}{Z1})(1 + \frac{R}{Z2})}
\]

(3.8)

where \( R \) is the 75 \( \Omega \) termination impedance.

For the preemphasis filter \( H_p(\omega) \), we have

\[
Z1 = \frac{R_L}{1 + j\omega C_p R_L} \quad (3.9)
\]

and

\[
Z2 = R3 + j\omega L_p \quad (3.10)
\]
Figure 3.4. Single horizontal scan of the unfiltered composite NTSC color bar pattern.
For the deemphasis filter $H_d(\omega)$, $Z1$ and $Z2$ are given by

$$Z1 = \frac{R1}{1 - j \frac{R1}{\omega L_d}}$$

(3.11)

and

$$Z2 = R3 - \frac{j}{\omega C_d}.$$  

(3.12)

The component values for $R1$, $R3$, $L_p$, $C_p$, $L_d$ and $C_d$ are given in Appendix 3. Figures 3.5 and 3.6 illustrate the gain and phase characteristics for the specified preemphasis and deemphasis networks. Figure 3.7 shows the placement of the preemphasis filter in the equivalent low pass phase error simulation. The baseband luminance signal $i(t)$ is filtered by the preemphasis network $H_p(\omega)$ whereas the equivalent low pass chrominance signal $\hat{c}(t)$ is filtered by the frequency shifted preemphasis filter $H_p(\omega + \omega_c)$.

**Integration**

The phase transfer function of a direct FM modulator (VCO) with a linear voltage to frequency transfer function is given as

$$I(\omega) = \frac{k_m}{j\omega},$$

(3.13)

where $k_m$ is the VCO conversion gain in radians/second/volt.

Equation 3.13 represents simple integration in the time domain. The preemphasized luminance signal $i_p(t)$ is filtered by $I(\omega)$ to yield the luminance phase input $\alpha(t)$ into the PLL error model. The equivalent low pass chrominance phase signal $\hat{\gamma}(t)$ is obtained by filtering the preemphasized chrominance signal $\hat{c}_p(t)$ by the frequency shifted integration function denoted by $I(\omega + \omega_c)$. The result is the phase input for the PLL error model operating in a linear mode.
Figure 3.5. Gain characteristics for the CCIR 405-1 FM preemphasis and deemphasis filters for a 525 line TVRO system.
Figure 3.6. Phase characteristics for the CCIR 405-1 FM preemphasis and deemphasis filters for a 525 line TVRO system.
Figure 3.7. Equivalent low pass, linear model of the FM transmitter (preemphasis plus direct FM modulation) and the phase-locked loop error model.
Figure 3.7 illustrates the placement of the integrator in the phase error simulation.

The value of $k_m$ was chosen to maintain the peak FM frequency deviation $\Delta f$ at 10.7 MHz which is standard for most satellite video transponders. By first computing the peak-to-peak value of the preemphasized composite video signal, $k_m$ can then be found by the following equation

$$k_m = \frac{4\pi \Delta f}{V_{p-p}},$$

(3.14)

where $V_{p-p}$ is the peak to peak amplitude of the video input to the FM modulator,

$\Delta f$ is the peak FM frequency deviation in Hz,

and $k_m$ is the VCO conversion gain in radians/second/volt.

The phase error can then be computed by applying the PLL error transfer function to the chrominance and luminance phase signals.

**Phase-Locked Loop Error Function**

For the error simulation, a second order PLL employing a lag-lead loop filter was simulated in the linear mode. Figure 3.8 illustrates the linear PLL model. The error transfer function $E(\omega)$ can be written as

$$E(\omega) = \frac{\phi_e(\omega)}{\theta(\omega)} = \frac{1}{k_d k_v \frac{1}{j\omega} F(\omega)},$$

(3.15)

where $k_d k_v$ is the open loop gain in seconds$^{-1}$.

For the lag-lead loop filter, $F(\omega)$ is defined as

$$F(\omega) = \frac{1 + j \frac{\tau_2}{2} \omega}{1 + j \tau_1 \omega},$$

(3.16)
Figure 3.8. Phase-locked loop linear model for small phase error.
where \( \tau_1 \) and \( \tau_2 \) are the pole and zero time constants for \( \text{F}(\omega) \) in seconds.

Substitution of equation 3.16 into equation 3.15 yields

\[
E(\omega) = \frac{-\omega^2 + j \frac{\omega}{\tau_1}}{\frac{k_d k_v}{\tau_1} - \omega^2 + j \omega \left( \frac{1 + k_d k_v}{\tau_1} \frac{\tau_2}{\tau_1} \right)}.
\]  
(3.17)

For the standard notation as applied to a linear second order response, the following substitutions are made:

\[
\omega_n^2 = \frac{k_d k_v}{\tau_1},
\]  
(3.18)

where \( \omega_n \) is the undamped natural frequency in radians/second,

and

\[
\xi = \frac{1 + k_d k_v}{2 \omega_n} \frac{\tau_2}{\tau_1},
\]  
(3.19)

where \( \xi \) is the loop damping coefficient.

Substitution of equations 3.18 and 3.19 into equation 3.17 yields

\[
E(\omega) = \frac{-\omega^2 + j \frac{\omega}{\tau_1}}{\omega_n^2 - \omega^2 + 2 j \xi \omega_n \omega}.
\]  
(3.20)

For the simulation, \( f_n \) (\( \omega_n / 2\pi \)) was considered as the independent parameter. The open loop gain \( k_d k_v \) was fixed at the value measured for the hardware phase error tests outlined in Section IV. The loop filter time constants \( \tau_1 \) and \( \tau_2 \) were adjusted for each value of \( f_n \) to keep \( \xi \) at 0.707 which provides a good trade off between PLL frequency response and stability. Figure 3.7 illustrates the placement of the PLL error function for the luminance and chrominance signals. The output of the error function \( E(\omega) \) is the luminance phase error \( \sigma(t) \).
and the output of the frequency shifted error function $E(\omega + \omega_c)$ is the equivalent low pass error $\tilde{\gamma}(t)$ due to the chrominance signal.

**RMS Phase Error Calculation**

As previously pointed out, the luminance signal $\ell(t)$ and chrominance signal $\tilde{\gamma}(t)$ are parallel processed by the system illustrated in Figure 3.7 to produce the luminance and chrominance error signals $\sigma(t)$ and $\tilde{\gamma}(t)$ respectively. The signals $\sigma(t)$ and $\tilde{\gamma}(t)$ can be expressed as the following Fourier series approximations which were used in the computer simulation:

$$
\sigma(t) = \sum_{n=0}^{N} \frac{k_m H_p(n\omega_h) E(n\omega_h) \ell_n}{j\omega_h} e^{jn\omega_h t}, \quad (3.21)
$$

where $\ell_n$ are the Fourier series coefficients for $\ell(t)$,

$$
\omega_h = 2\pi f_h,
$$

$N$ is the physical size of the complex data file for $\ell(t)$ and $\tilde{\gamma}(t)$ ($N = 1024$ complex data points);

and

$$
\tilde{\gamma}(t) = \sum_{n=0}^{N} \frac{k_m H_p(n\omega_h + \omega_c) E(n\omega_h + \omega_c) \tilde{c}_n}{j(n\omega_h + \omega_c)} e^{jn\omega_h t}, \quad (3.22)
$$

where $\tilde{c}_n$ are the Fourier series coefficients for $\tilde{\gamma}(t)$.

The composite error signal $\varepsilon(t)$ can now be expressed in terms of $\sigma(t)$ and $\tilde{\gamma}(t)$ by

$$
\varepsilon(t) = \sigma(t) + |\tilde{\gamma}(t)| \cos[\omega_c t + \angle \tilde{\gamma}(t)]. \quad (3.23)
$$

For the purposes of calculating the RMS value of equation 3.23, it was first necessary to compute the time domain coefficients for the chrominance and luminance phase error signals ($\tilde{\gamma}$ and $\sigma$ respectively) by application of an inverse discrete Fourier transform. The RMS value of $\varepsilon(t)$ is thus found to be
\[ E_{\text{RMS}} = \sqrt{\frac{1}{N} \sum_{n=1}^{N} \left( \frac{1}{2} |\tilde{\gamma}_n|^2 + |\sigma_n|^2 \right)} \]  \hspace{1cm} (3.24)

where \( \tilde{\gamma}_n \) and \( \sigma_n \) are the discrete time coefficients for \( \tilde{\gamma}(t) \) and \( \sigma(t) \) respectively, and \( E_{\text{RMS}} \) is the PLL RMS phase error in radians.

Section IV contains the data obtained for the RMS phase error for selected values of the loop natural frequency \( f_n \). The computer generated data is compared with the experimental results for the same values of \( f_n \). A description of the hardware implementation is also given in Section IV and Appendix 3.

**Signal Analysis Routines**

To verify the proper operation of the software simulation and to obtain a better understanding for the system behavior, several algorithms were written to allow the creation of data files representative of various time and frequency domain signals. These signals of interest include: the composite video signal, the log magnitude spectrum of the video signal, the PLL input phase signal, the log magnitude FM spectrum and the composite demodulated video signal. Example plots of these signals along with their respective experimental measurements are given in Section V.

The time domain composite video signal \( v(t) \) is found by equation 3.7. The preemphasized video signal \( v_p(t) \) is found by application of equation 3.7 on the preemphasized chrominance and luminance signals \( \tilde{c}_p(t) \) and \( \ell_p(t) \) respectively. In both cases, the resulting data file for \( v(t) \) or \( v_p(t) \) contains 2048 real data points which can be plotted.
using an external graphics routine (see software listing in Appendix 1).

The single sided, log magnitude frequency spectrum for \( v(t) \) or \( v_p(t) \) is found by plotting the log magnitude of the positive frequency Fourier coefficients for the luminance signal \( \hat{x}_n \) along with the log magnitude of the equivalent low pass chrominance coefficients \( \hat{c}_n \) centered on the chrominance subcarrier \( f_c \). To better represent the discrete nature of the video frequency spectra, the number of data points needed was doubled for which every other data point was set to a specified noise floor. The result is a very accurate representation of the video spectrum (see Section V, Figure 5.5). To represent the time domain phase input \( \theta(t) \) (radians) into the PLL, the algorithm shown in Figure 3.9 was employed. The inputs are the frequency domain signals representing the chrominance phase \( \hat{\phi}(\omega) \) and the luminance phase \( \hat{\alpha}(\omega) \). A 5-pole Butterworth lowpass filter is applied to the 4096 point \( \theta(t) \) file to eliminate unwanted harmonics above the Nyquist frequency of approximately 8 MHz. Figure 3.9 also illustrates the algorithm for obtaining the FM spectra \( |\hat{s}(\omega)| \) from \( \theta(t) \). The \( |\hat{s}(\omega)| \) data file also contains 4096 real data points.

The demodulated video signal \( m(t) \) can be obtained by referring to the PLL linear model of Figure 3.8. The error signal \( \phi_e(t) \) is filtered by the loop filter \( F(\omega) \) to obtain \( m(t) \). The phase detector gain factor \( k_d \) is disregarded since we are only concerned with the relative content of \( m(t) \) as it compares with the original signal \( v(t) \). Deemphasis \( [H_d(\omega)] \) is also included to restore the original video.
Figure 3.9 Method of obtaining the time domain PLL phase input $\theta(t)$ and the magnitude of the FM spectra $|\tilde{s}(\omega)|$. 

\[ \alpha(w) \xrightarrow{\text{IDFT}} \alpha(t) \xrightarrow{\text{REAL}(\cdot)} \sum \xrightarrow{\text{Low Pass Filter}} \theta(t) \text{ Phase} \]

\[ \tilde{\beta}(w) \xrightarrow{\text{IDFT}} \tilde{\beta}(t) \xrightarrow{e^{jwt}} \text{REAL}(\cdot) \xrightarrow{\sum} \theta(t) \text{ Phase} \]

\[ e^{j(\cdot)} \xrightarrow{\text{DFT}} |\cdot| \xrightarrow{\text{FM Spectra}} |\tilde{s}(\omega)| \]
signal. The resulting outputs as shown in Figure 3.10 are the demodulated chrominance ($\hat{c}(t)$) and luminance ($\hat{l}(t)$) signals to which equation 3.7 can be applied to obtain the composite signal $m(t)$. 
Figure 3.10. Equivalent low pass model for obtaining the PLL demodulated signal.
IV. Experimental Measurement of Phase Error

The major components used in the hardware phase error measurement system are shown in Figure 2.1. Schematics and parts lists for the NTSC color bar generator, the CCIR 405-1 preemphasis filter, the FM modulator and the PLL demodulator are given in Appendix 3.

Before making the phase error measurements, the chrominance levels of the NTSC color bar generator were adjusted for 100% saturated color bars. The peak-to-peak output of the generator was also adjusted to maintain the peak FM frequency deviation $\Delta f$ of the FM modulator at 10.7 MHz. The measured conversion gain for the modulator was found to be

$$k_m = 79.3 \text{ MHz/volt.}$$

Using equation 3.14 with $k_m$ in units of radians/second/volt, we obtain the peak-to-peak video level $V_{p-p}$ for the FM modulator input which is found to be

$$V_{p-p} = 270 \text{ mV.}$$

To make the signal levels from the FM modulator output to the PLL input compatible, it was necessary to place a 20 dB attenuator between the two system components. The signal level into the PLL was maintained at $30 \text{ mV}_{\text{RMS}}$ to insure that the PLL phase detector gain $k_d$ was kept at a maximum.

The phase detector gain $k_d$ of the PLL was measured by free running the PLL VCO while applying a 70 MHz CW signal at the phase detector input. Refering to the PLL schematic in Appendix 3, jumper 1 was placed in the 1 to 2 position and R100 (Free Run Adj.) was adjusted to maintain a 2 MHz sinewave at pin 11 of the phase detector.
(IC101). The peak value of this sinewave indicates the phase detector gain $k_d$ which was measured as

$$k_d = 0.125 \text{ volts/rad.}$$

The gain $k_v$ for the VCO of the PLL was measured to include the conversion gain of the VCO chip (IC100) and the gain of the loop amplifier which consists of transistors Q100 and Q101. The measured value of $k_v$ was

$$k_v = 294 \text{ MHz/volt}$$

or

$$k_v = 1.848 \times 10^9 \text{ radians/second/volt.}$$

The open loop gain of the PLL is simply

$$k_d k_v = 321 \times 10^6 \text{ seconds}^{-1}.$$ 

The above value for $k_d k_v$ is fixed for both the hardware tests and software simulations.

To measure the phase error of the PLL, the loop was first closed by placing jumper 2 in the 2 to 3 position. Jumper 2 was placed in the 2 to 3 position to allow the phase error signal to be measured at the composite video output (J102) of the demodulator board. The video filter between Q103 and Q104 is required to eliminate the presence of 70 MHz carrier which would otherwise result in erroneous data. The 3 dB cutoff of the filter is approximately 8 MHz which allows a flat response over the video bandwidth of 0 to 4.2 MHz.

The gain $G_p$ for the signal path from the phase detector output (pin 11 of IC101) to the composite video output (J102) was measured as

$$G_p = 0.893 \frac{\text{volts}}{\text{volt}}.$$ 

After measuring the true RMS value ($V_{RMS}$) of the composite video output, the RMS phase error is then found by
\[ E_{RMS} = \frac{V_{RMS}}{G_p k_d} \]  

(4.1)

where \( k_d \) is in units of volts/radian.

The true RMS value of the video signal was measured using a Fluke model 910A 10 MHz true RMS meter. The resulting RMS phase error is found from equation 4.1.

The four major components used in the hardware phase error tests are shown in Figure 4.1. Figure 4.2 illustrates the complete lab measurement setup including the true RMS meter and a oscilloscope plus spectrum analyzer for monitoring the signals at various points in the system.
Figure 4.1 The four major components employed in the experimental phase error measurements.

Figure 4.2 Lab setup for the experimental phase error measurements.
V. Comparison of Measured and Simulated System Performance

To verify that a good correlation exists between software generated and hardware implemented signals, the following series of illustrations (Figure 5.1 to 5.7) are included. For each case, $f_n$ is maintained at 15 MHz. The remaining system parameters are summarized in Table 6.1.

In addition, Figure 5.8 illustrates the PLL phase input signal $\theta(t)$ for the parameter values indicated. Figure 5.9 is a sample of the demodulated NTSC test signal $m(t)$. The demodulated signal, for most values of $f_n$, was found to be almost exactly correlated with the original video signal $v(t)$.
Figure 5.1 Measured time domain video signal with preemphasis (Top) and without preemphasis (Bottom).

Horizontal: 10 µsec./div.
Vertical: 0.1 V/div.
Figure 5.2. Simulated NTSC color bar test signal with chrominance and luminance low pass filters applied.
Figure 5.3. Simulated NTSC test signal with FM preemphasis.
Figure 5.4 Measured video spectrum for the preemphasized NTSC color bar pattern.

Horizontal: 0.5 MHz/div., 2.5 MHz center
Vertical: 10 dB/div.
Figure 5.5. Simulated spectrum of the preemphasized NTSC test signal.
Figure 5.6 Measured FM spectrum for the NTSC test signal with $\Delta f = 10.7$ MHz.

Horizontal: 5 MHz/div., 70 MHz center
Vertical: 10 dB/div.
Figure 5.7. Simulated F1 spectra for the NTSC test signal with $f_0 = 10.7$ MHz.

Relative Amplitude (dB) vs. Frequency (MHz)
Figure 5.3. Simulated time domain PLL phase input.
Figure 5.9. Example of a demodulated test signal with deemphasis applied.
VI. Phase Error Results

Table 6.1 summarizes the values of the fixed system parameters based on the hardware described in Section IV or the standards set forth by NTSC and CCIR. The nine values of \( f_n \) chosen for phase error testing range from 5 MHz to 25 MHz and are spaced on an interval of 2.5 MHz. To make the hardware PLL parameters compatible with the software parameters, it was necessary to compare the theoretical and measured closed loop demodulator frequency response curves for each value of \( f_n \). Refering to Figure 3.8, the loop frequency response \( D(\omega) \) is defined as

\[
D(\omega) = \frac{\dot{\theta}(\omega)}{\dot{\theta}(\omega)}. \tag{6.1}
\]

Figure 6.1 illustrates the theoretical frequency response curves for the values of the natural frequency previously mentioned. By adjusting the hardware PLL parameters (namely \( \tau_1 \) and \( \tau_2 \)), the frequency response of the PLL can be made to match the theoretical response curve for each value of the loop natural frequency. Referring to the PLL demodulator schematic of Appendix 3, the values of \( \tau_1 \) and \( \tau_2 \) can be varied by adjusting the loop filter components R103 and C108. It should be noted that for small values of \( f_n \) (namely 5 MHz and 7.5 MHz), it was necessary to place additional capacitance in parallel with C108 and additional resistance in series with R103 to obtain the desired loop response. By first matching the theoretical and measured loop responses as closely as possible, a pair of phase error data points could then be taken for a given natural frequency.

Figure 6.2 illustrates the results of the aforementioned phase error tests. It is interesting to note that previous calculations and assumptions made prior to this report assumed a range of 20 MHz to 25
Table 6.1 Summary of the fixed system parameters for the experimental and simulated phase error measurements.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Description</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>(PERSAT)</td>
<td>Percentage color saturation for the NTSC color bar pattern</td>
<td>100</td>
<td>%</td>
</tr>
<tr>
<td>(YPOL)</td>
<td>The number of poles for the Butterworth video filter $H_v(\omega)$</td>
<td>3</td>
<td>--</td>
</tr>
<tr>
<td>(QPOL)</td>
<td>The number of poles for the Q chrominance filter $H_q(\omega)$</td>
<td>3</td>
<td>--</td>
</tr>
<tr>
<td>(IPOL)</td>
<td>The number of poles for the I chrominance filter $H_i(\omega)$</td>
<td>3</td>
<td>--</td>
</tr>
<tr>
<td>(YCUT)</td>
<td>The 3 dB cutoff frequency of $H_v(\omega)$</td>
<td>4.2</td>
<td>MHz</td>
</tr>
<tr>
<td>(QCUT)</td>
<td>The 3 dB cutoff frequency of $H_q(\omega)$</td>
<td>0.5</td>
<td>MHz</td>
</tr>
<tr>
<td>(ICUT)</td>
<td>The 3 dB cutoff frequency of $H_i(\omega)$</td>
<td>1.5</td>
<td>MHz</td>
</tr>
<tr>
<td>$\Delta f$</td>
<td>The peak FM frequency deviation of the FM modulator</td>
<td>10.7</td>
<td>MHz</td>
</tr>
<tr>
<td>(FMPKDEV)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\xi$</td>
<td>The loop damping coefficient for the PLL linear model</td>
<td>0.707</td>
<td>--</td>
</tr>
<tr>
<td>(DAMP)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$k_d k_v$</td>
<td>The open loop gain of the PLL</td>
<td>$231 \times 10^6$</td>
<td>sec$^{-1}$</td>
</tr>
<tr>
<td>(KDKV)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: The parameter names used in the software simulation are given in ( ).
Figure 6.1. Theoretical PLL demodulator frequency response curves for the nine tested values of $f_n$. 
Figure 6.2. RMS phase error data for the software simulation and the hardware tests.
MHz as an ideal range of natural frequency. Figure 6.2, however, indicates that a natural frequency as low as 10 MHz would maintain a small phase error while keeping the noise bandwidth low enough for satellite FM demodulation of a signal with a relatively low signal-to-noise ratio. Additional observations indicate that for low values of $f_n$ ($f_n \leq 7.5$ MHz), the resulting increased phase error caused severe distortion in the demodulated signal. After testing the demodulator in a satellite receiver (see Section VII), it was observed that loop natural frequencies greater than 20 MHz caused a noticeable decrease in the demodulator output signal-to-noise ratio due to high PLL noise bandwidth. In the context of this experiment, the data taken and the qualitative observations made, the author has reached the general conclusion that a loop natural frequency of 10 MHz to 20 MHz is best for a second order PLL FM video demodulator. For the satellite receiver briefly mentioned in the following section, a natural frequency of $f_n = 15$ MHz was chosen and found to perform very well.
VII. Satellite FM Receiver

To supplement the work documented in this paper, it was the author's intent to build a satellite receiver to allow a "real world" application for the PLL constructed for the hardware phase error tests. Such a receiver was designed and constructed in the fall of 1985 and spring of 1986. The block diagram of the receiver is shown in Figure 7.1. As well as the PLL video demodulator, the receiver also includes a 70 MHz IF prefilter, an analog video processor board, a dual PLL stereo demodulator and a DC control and metering card. A more detailed description of the receiver operation and alignment procedures is outlined in a separate manual (PLL Satellite Receiver Technical Manual) which supplements this report. The receiver proved to be an interesting and informative tool for verifying the performance of the PLL in the environment for which it was originally intended to be used.
VIII. Conclusions and Future Suggested Studies

The entire study documented in this report has proved to be an interesting, informative and worthwhile topic of research. The conclusions, though somewhat limited, provide invaluable information for the design of a second order PLL video demodulator. The author suggests future studies in the same area which would include such factors as the audio subcarriers, 70 MHz IF prefiltering, nonlinearities due to satellite FM transmission and the nonlinearity of the PLL phase detector.

An additional proposed study would make use of the satellite receiver to compare the PLL demodulator to an FM quadrature detector which is currently a popular scheme for FM video demodulation. Regardless of the details of any future work, it is hoped that studies in the area of satellite transmission and reception will continue at this institution due to their popularity in the field of electronic communications.
IX. Bibliography


D. Hummels, *Analysis of Communication Systems*, EE690 Class Notes, Kansas State University, Manhattan, Kansas, 1981.


Appendix 1. Software Listing
INITIATING SEQUENCE:
RUN NTSC

PURPOSE:

This program is intended to study the rms phase error of a second order phase locked loop employed as a satellite FM video demodulator. The program consists of three major blocks which are:

1) NTSC color bar pattern generator with CCIR 405-1 preemphasis for satellite FM transmission.
2) FM transmitter phase simulation (integration)
3) Linear phase locked loop simulation with rms phase error computation

Each program segment contains menu selections for user input of system parameters and desired operations. The user also has the option of generating data files for the following signals which can be displayed using any graphics package employing the SGOPEN and SGTRAN data transfer routines:

1) Time domain video signal (single horizontal scan)
2) Video spectra (log magnitude data)
3) Time domain pll phase input signal (radians)
4) FM spectra (log magnitude data)
5) Time domain demodulated video signal

The subroutines for the NTSC program are located in the following files:

1) NTSCAUX.FOR (Auxiliary subroutine such as data storage and retrieval, menu generators, parameter storage and retrieval, etc.)
2) NTSCFIL.FOR (DFT and frequency domain filter subroutines)
3) NTSCGEN.FOR  (NTSC signal generating subroutines)

The common area of variables and data arrays is located in:

NTSCCOM.FOR

The NTSC generator and FM modulator/PLL demodulator default parameters are located in:

NTSCPARA.TXT

After compilation, the .FOR files can be linked by the following VMS command:

LINK NTSC,NTSCAUX,NTSCFIL,NTSCGEN

ROUTINE(S) ACCESSED OR CALLED BY THIS ROUTINE:

NTSCAUX SUBROUTINES:

MENGST  (Menu for NTSC signal generator or NTSC data retrieval)
MENGEN  (Menu for editing the NTSC generator parameters)
MENNXT1 (Menu for selecting the next action after the NTSC generator)
MENFMPLL (Menu for editing the FM modulator and PLL demodulator parameters)
MENNXT2 (Menu for selection of the next action after the PLL demodulator and phase error calculation)
GETPARA  (System default parameter retrieval routine)
SHOWPARA (System parameter display routine)
STOREPARA (System parameter storage routine)
FIGET    (Data retrieval routine)
FISTORE  (Data storage routine)
WAIT_RET (Wait for <RETURN> to continue program)
DATSTAT  (Returns the maximum, minimum and average of a real data array)
PHAERROR (Calculates the RMS phase error from the complex PLL error DFT coefficients)

NTSCGEN SUBROUTINES:

NTSCGEN  (Generates the DFT luminance and chrominance coefficients for the NTSC color bar signal)
TIMEVIDEO (Generates the time domain NTSC video signal)
FREQVIDEO (Generates the log magnitude spectra of the NTSC video signal)
TIMEPHASE (Generates the time domain PLL input phase signal in radians)
FMSPEC (Generates the log magnitude FM spectra of the PLL input)
PLLDEMOD (Generates the time domain demodulated video signal)

NTSCFIL SUBROUTINES:
DFT (Discrete Fourier and inverse Fourier transform for complex data arrays)
INTEG (Frequency domain integration for complex data arrays)
BWLPF (Butterworth low pass filter for complex data arrays)
PREEMPH (CCIR 405-1 FM preemphasis and deemphasis for complex data arrays)
ERRFUNC (Linear PLL error function for complex data arrays)

C***********************************************************************************************
C***********************************************************************************************

BEGIN THE MAIN ROUTINE

C***********************************************************************************************
C***********************************************************************************************

INCLUDE 'NTSCCOM.FOR'
CHARACTER*15 FINAME
CHARACTER*1 RESPONSE
INTEGER NUM,CHOICE
LOGICAL PHAFLG

C***********************************************************************************************
C
C INITIALIZE ALL CONSTANTS FOR THE NTSC ROUTINE

C***********************************************************************************************
C
PI = 3.141593

Set horizontal line frequency FH & WH (Hz & rad/sec)
FH = 15734.26
WH = 2.0 * PI * FM

C Set color subcarrier frequency FC & WC (Hz & rad/sec)
FC = 3579545.0
WC = 2.0 * PI * FC

C Set the video max and min variables to 0
MAX = 0.0
MIN = 0.0

C Set the default parameter values
CALL GETPARA

******************************************************************************
C CALL THE MENU TO SELECT THE SOURCE OF THE NTSC DFT COEFFICIENTS
C******************************************************************************
100 CALL MENGET(CHOICE)

******************************************************************************
C MENU SELECTION #1: GENERATE THE NTSC SIGNAL
C******************************************************************************

IF (CHOICE.EQ.1) THEN

C Display the menu for the ntsc generator routine parameters
C and let the user select the desired parameters
CALL MENGEN

C Generate the ntsc dft coefficients
STATUS = LIBERASE_PAGE(1,1)
TYPE*, "GENERATING THE NTSC COEFFICIENTS"
CALL NTSCGEN

C Prompt the user for storage of the coefficients
TYPE*, "STORE THE NTSC GENERATOR COEFFICIENTS? (Y/N) ----> ",S
ACCEPT S, RESPONSE

111 FORMAT (T2,'STORE THE DFT COEFFICIENTS? (Y/N) ----> ',S)

112 FORMAT (A1)

IF (RESPONSE.EQ.'Y'. OR RESPONSE.EQ.'y') THEN

TYPE*, "STORE THE LUMINANCE DFT COEFFICIENTS (L(w))"
CALL FIFOSTORE ('LUM.DAT',N,'COMPLEX',LUM)

TYPE*, "STORE THE CHROMINANCE DFT COEFFICIENTS (C-(w))"
CALL FIFOSTORE ('CHR.DAT',N,'COMPLEX',CHR)

CALL WAIT_RET
END IF
END IF

******************************************************************************
C
C MENU SELECTION #2: RECALL THE NTSC COEFFICIENTS
C
IF (CHOICE.EQ.2) THEN
C
Recall a previously stored set of dft coefficients
TYPE*, ' ' 
TYPE*, 'RECALL THE LUMINANCE DFT COEFFICIENTS:'
CALL FIGET ('LUM.DAT', NUM, 'COMPLEX', LUM)
TYPE*, ' ' 
TYPE*, 'RECALL THE CHROMINANCE DFT COEFFICIENTS:'
CALL FIGET ('CHR.DAT', NUM, 'COMPLEX', CHR)
CALL WAIT RET
END IF
C
C******************************************************************************
C
C CALL THE MENU TO SELECT THE NEXT DESIRED ACTION
C
C******************************************************************************
200  STATUS = LIB$ERASE_PAGE(1,1)
     CALL MENNXT1 (CHOICE)

C
C MENU SELECTION #1: GENERATE THE TIME DOMAIN VIDEO SIGNAL
C
C******************************************************************************
C
IF (CHOICE .EQ. 1) THEN
   STATUS = LIB$ERASE_PAGE(1,1)
   TYPE*, 'GENERATING THE TIME DOMAIN VIDEO SIGNAL'
   CALL TIMEVIDEO
C
   Find the max and min of the video signal
   CALL DATSTAT (VIDEOSIGNAL, N2, MAX, MIN, AVE)
C
   Store the time domain signal
   TYPE*, ' ' 
   TYPE*, 'STORE THE TIME DOMAIN VIDEO SIGNAL'
   CALL FISTORE ('TIMEVIDEO.DAT', N2, 'REAL', VIDEOSIGNAL)
C
   Return to the current menu
   CALL WAIT RET
   GOTO 200
END IF

C
C MENU SELECTION #2: GENERATE THE VIDEO SPECTRUM
C
C******************************************************************************
C
IF (CHOICE .EQ. 2) THEN
C
Enter the noise floor for the dft plot
STATUS = LIB$ERASE_PAGE(1,1)
TYPE*, 'ENTER THE NOISE FLOOR FOR THE NORMALIZED LOG SPECTRA'
TYPE114
114  FORMAT (T2, 'THE DEFAULT IS -80.0 dB ------> ',S)
     ACCEPT130, NOISEFL
FORMAT (F10.3)
IF (NOISEFL .EQ. 0.0) NOISEFL = -80.0

C Generate the video spectra
    TYPE*, 'GENERATING THE VIDEO SPECTRA'
    CALL FREQVIDEO

C Store the video spectra
    TYPE*, 'STORE THE VIDEO SPECTRA'
    CALL FSTORE ('FREQVIDEO.DAT', N2, 'REAL', VIDEOSPECTRA)

C Return to the current menu
    CALL WAIT_RET
    GOTO 200
END IF

C*******************************************************************************
C
C MENU SELECTION #3: STORE THE CURRENT NTSC PARAMETERS
C*******************************************************************************
C
    IF (CHOICE .EQ. 3) THEN
        CALL STOREPARAM
        GOTO 200
    END IF

C*******************************************************************************
C
C MENU SELECTION #4: RETURN TO THE FIRST MENU
C*******************************************************************************
C
    IF (CHOICE .EQ. 4) GOTO 100

C*******************************************************************************
C
C MENU SELECTION <RETURN>: CONTINUE WITH THE SIMULATION
C*******************************************************************************
C
C*******************************************************************************
C
C DISPLAY THE MENU FOR SELECTION OF THE FM MODULATION AND DEMODULATION C PARAMETERS
C*******************************************************************************
C
    STATUS = LIBSREASE_PAGE(1,1)
    TYPE*, 'BEGIN FM MODULATOR / DEMODULATOR SIMULATION'

C Calculate the max and min of the current video signal
    CALL TIMEVIDEO
    CALL DATSTAT ('VIDEO SIGNAL', N2, MAX, MIN, AVE)

C Calculate the fm modulation constant km in mHz per volt
    KM = 2.0 * FMKDEV / (MAX - MIN)

C Calculate the dependent loop parameters
    WN = SQRT (KDKV / (T1 * 1.0E-9))
    FN = WN / (2.0 * PI * 1.0E6)
DAMP = 0.5 * (1 + KDKV * T2 * 1.0E-9) / SQRT (KDKV * T1 * 1.0E-9)

CALL MENMFPLL

*******************************************************************************
C CALCULATE THE RMS PHASE ERROR
C*******************************************************************************

Integrate the dft coefficients for the luminance and chrominance signals to obtain the phase coefficients. Set the dc term of the signal to zero and multiply the coefficients by KM (rad/sec/volt) before the integration.

DO I = 1,N
   PHALUM(I) = CMPLX (KM * 2.0 * PI * 1.0E6, 0.0) * LUM (I)
   PHACHR(I) = CMPLX (KM * 2.0 * PI * 1.0E6, 0.0) * CHR (I)
END DO

Set the dc value to 0 before integration
PHALUM (1) = (0.0 , 0.0)

CALL INTEG (PHALUM, N, FH, 0.0)
CALL INTEG (PHACHR, N, FH, FC)

Implement the error transfer function of the pll to obtain the phase error coefficients

DO I = 1,N
   ERRLUM (I) = PHALUM (I)
   ERRCHR (I) = PHACHR (I)
END DO

CALL ERFUNC (ERRLUM, N, FH, 0.0, DAMP,FN,T1)
CALL ERFUNC (ERRCHR, N, FH, FC, DAMP,FN,T1)

Compute the phase error
CALL PHAERROR

Display the phase error

SET*,'RMS PHASE ERROR = ',RMSERROR,' radians = ',RMSERROR * 180.0 / PI,' degrees'

Set the time domain phase flag to .FALSE. indicating that the new phase signal has not yet been generated
PHAFLAG = .FALSE.

CALL WAIT_RET

*******************************************************************************
C DISPLAY THE MENU FOR THE NEXT ACTION
C*******************************************************************************

CALL MENNXT2 (CHOICE)
C MENU SELECTION #1: COMPUTE THE TIME DOMAIN PHASE SIGNAL
C
C-------------------------------------------------------------------------------------
C IF (CHOICE .EQ. 1) THEN

C Set the phase flag to .TRUE. indicating that a new phase signal has
C been generated
PHAFLG = .TRUE.

C Generate the phase signal
STATUS = LIB$ERASE_PAGE(1,1)
TYPE*, 'GENERATING THE TIME DOMAIN PLL INPUT PHASE SIGNAL'
CALL TIMEPHASE

C Store the phase signal
TYPE*, 'STORE THE PHASE SIGNAL'
CALL FISTORE ("PHASE.DAT",N4,'REAL',PHASE)

C Return to the current menu
CALL WAIT_RET
GOTO 500
END IF

C-------------------------------------------------------------------------------------
C MENU SELECTION #2: COMPUTE THE FM SPECTRA
C
C-------------------------------------------------------------------------------------
C IF (CHOICE .EQ. 2) THEN

C If the phase signal has already been generated, then there is no
C need to generate it again.
STATUS = LIB$ERASE_PAGE(1,1)

IF (NOT(PHAFLG)) THEN
  TYPE*, 'GENERATING THE TIME DOMAIN PLL INPUT PHASE SIGNAL'
  CALL TIMEPHASE
END IF

C Enter the noise floor for the FM spectra
TYPE*, 'ENTER THE NOISE FLOOR FOR THE NORMALIZED LOG SPECTRA'
TYPE111
131 FORMAT (T2,'THE DEFAULT IS -60.0 dB -------> ',S)
ACCEPT132, NOISEFL
132 FORMAT (F10.3)
IF (NOISEFL .EQ. 0.0) NOISEFL = -60.0

C Generate the FM spectra
TYPE*, 'GENERATING THE FM SPECTRA'
CALL FMSPEC

C Store the FM spectra
TYPE*, 'STORE THE FM SPECTRA'
CALL FISTORE ("FMSPECTRA.DAT",N4,'REAL',FMSPECTRA)

C Return to the current menu
CALL WAIT_RET
GOTO 500
END IF

C
C MENU SELECTION #3: COMPUTE THE DEMODULATED SIGNAL
C
C IF (CHOICE .EQ. 3) THEN
C Compute the demodulated signal
STATUS = LIB$ERASE_PAGE(1,1)
TYPE*, 'COMPUTING THE DEMODULATED SIGNAL'
CALL PLLDEM0D
C Store the demodulated signal
TYPE*,''
TYPE*, 'STORE THE DEMODULATED SIGNAL'
CALL FISTORE ('DEM0.DAT',N2,'REAL',DEMOD)
C Return to the current menu
CALL WAIT_RET
GOTO 500
END IF

C
C MENU SELECTION #4: RETURN TO THE FM MODULATOR / PLL PARAMETER MENU
C
C IF (CHOICE .EQ. 4) GOTO 400

C
C MENU SELECTION #5: GO BACK TO THE NTSC GENERATOR MENU
C
C IF (CHOICE .EQ. 5) GOTO 100

C
C MENU SELECTION #6: STORE THE CURRENT NTSC PARAMETERS
C
C IF (CHOICE .EQ. 6) THEN
CALL STOREPARA
GOTO 500
END IF

C
C MENU SELECTION #7: RECALL NTSC PARAMETERS FROM DISK
C
C IF (CHOICE .EQ. 7) THEN
CALL GETPARA
GOTO 500
END IF
MENU SELECTION #8: SHOW THE NTSC PARAMETERS STORED ON DISK

IF (CHOICE .EQ. 8) THEN
    CALL SHOWPARA
    GOTO 500
END IF

MENU SELECTION <RETURN>: STOP THE PROGRAM

STOP
END

******************************************************************************
VAX-11 FORTRAN SOURCE FILENAME: NTSCOM.FOR
DEPARTMENT OF ELECTRICAL ENGINEERING KANSAS STATE UNIVERSITY
REVISION DATE PROGRAMMER(S)
---------- ------ ----------------
3.0 4/2/86 DALE DUBBERT

******************************************************************************
PURPOSE:
COMMON AREA FOR NTSC GENERATOR AND PLL SIMULATOR ROUTINE

******************************************************************************
DESCRIPTION OF VARIABLES USED IN THE MAIN ROUTINE NTSC.FOR:

DATA ARRAYS:

LUM  The DFT coefficients of the video luminance signal l(w) (complex)
CHR  The DFT coefficients of the equivalent low pass chrominance signal c-(w) (complex)
PHALUM  The DFT coefficients of the luminance phase input of the phase locked loop
PHACHR  The DFT coefficients of the equivalent low pass chrominance phase input of the phase locked loop (complex)
ERRLUM  The DFT coefficients of the luminance PLL phase error (complex)
ERRCHR  The DFT coefficients of the equivalent low pass chrominance PLL phase error (complex)
DELMUM  The DFT coefficients of the demodulated luminance signal (complex)
DEMCHR  The DFT coefficients of the equivalent low pass demodulated chrominance signal (complex)
CBUFFER  Temporary data buffer (complex)
RBUFFER  Temporary data buffer (real)
VIDEOSIGNAL  The NTSC color bar composite video signal (real)
VIDEOSPECTRA  The log magnitude spectra of the NTSC video signal (real)
PHASE  The time domain PLL input phase signal in radians (real)
FMSPECTRA  The log magnitude spectra of the PLL FM input (real)
DEMOD  The time domain PLL demodulated signal (real)
RMSERROR  The RMS value of the PLL error signal (real)

NTSC GENERATOR PARAMETERS:
YPOL  The number of poles for the luminance low pass filter
QPOL  The number of poles for the Q chrominance low pass filter
IPOL  The number of poles for the I chrominance low pass filter
YCUT  The cutoff frequency (MHz) for the luminance LPF
QCU T  The cutoff frequency (MHz) for the Q chrominance LPF
ICUT  The cutoff frequency (MHz) for the I chrominance LPF
PREFLG  The 'Y' or 'N' flag for requesting preemphasis or no preemphasis
PERSAT  The percent color saturation for the NTSC color bar signal
NOISEFL  The noise floor (dB) for log magnitude plots

FM MODULATOR AND PLL DEMODULATOR PARAMETERS:
FMPKDEV  The peak deviation (MHz) for the FM signal
MAX  The maximum voltage of the FM transmitter input
MIN  The minimum voltage of the FM transmitter input
AVE  The average (dc) voltage of the FM transmitter input
The modulation index for the FM transmitter (MHz/volt)
The open loop gain of the phase locked loop (1/seconds)
The natural frequency of the phase locked loop (radians/second)
The natural frequency of the phase locked loop (MHz)
The damping coefficient of the phase locked loop
The PLL loop filter pole time constant (nanoseconds)
The PLL loop filter zero time constant (nanoseconds)

The horizontal scan frequency for a 525 line television system (Hz)
The horizontal scan frequency for a 525 line television system (radians/second)
The color subcarrier frequency for a 525 line television system (Hz)
The color subcarrier frequency for a 525 line television system (radians/second)

Data arrays:
- COMMON /DATA/ LUM, CHR, PHALUM, PHACHR, ERRLUM, ERRCHR, RBUFFER, VIDEOSIGNAL, VIDEOSPECTRA, PHASE, FMSPECTRA, GBUFFER, FMERROR, DEMOD, DEMLUM, DEMCHR

NTSC signal generator parameters:
- COMMON /NTSCPAR/ YPOL, QPOL, IPOL, YCUT, QCUT, ICUT, PREFLG, PERSAT, NOISEFL

FM transmitter and PLL demodulator parameters:
- COMMON /PLLPAR/ FMPKDEV, MAX, MIN, AVE, KM, KDKV, WN, FN, DAMP, T1, T2

System constants:
- COMMON /CONST/ FH, FC, WH, WC, PI, PO

DATA DECLARATION

INTEGER N, N2, N4, N8, N16, I, YPOL, QPOL, IPOL
INTEGER*4 STATUS

PARAMETER (N=1024,N2=2048,N4=4096,N8=8192,N16=16384)

REAL RBUFFER(N4), FH, FC, WH, WC, PI, PO, YCUT, QCUT, ICUT,
   & PERSAT, VIDEOSIGNAL(N2), VIDEOBANZ(N2), NOISEFL,
   & FMPKDEV, MAX, MIN, AVE, KM, KDKV, WN, FN, DAMP, T1, T2,
   & PHASE(N4), FMSPECTRA(N4), RMSERROR, DEMOD(N2)

CHARACTER*1 PREFLG

COMPLEX LUM(N), CHR(N), PRELUM(N), PRECHR(N), PHALUM(N), PHACHR(N),
   & ERRLUM(N), ERRCHR(N), DEMLUM(N), DEMCHR(N), ARG,
   & CBUFFER(N4)

**********************************************************************

VAX-11 FORTRAN SOURCE FILENAME: NTSCGEN.FOR

DEPARTMENT OF ELECTRICAL ENGINEERING KANSAS STATE UNIVERSITY

REVISION DATE PROGRAMMER(S)

3.0 4/2/86 DALE DUBBERT

PROJECT: MS REPORT: RMS PHASE ERROR OF A SECOND ORDER PHASE
LOCKED LOOP EMPLOYED AS A SATELLITE FM VIDEO DEMODULATOR

**********************************************************************

RMS PHASE ERROR SIMULATION SIGNAL GENERATOR ROUTINES:

NTSCGEN  {NTSC COLOR BAR PATTERN GENERATOR}
TIMEVIDEO {TIME DOMAIN VIDEO SIGNAL GENERATOR}
FREQVIDEO {VIDEO SPECTRA GENERATOR}
TIMEPHASE {PLL INPUT PHASE SIGNAL GENERATOR}
FMSPEC {PLL FM INPUT SPECTRA GENERATOR}
PLLDEMOD {PLL DEMODULATED SIGNAL GENERATOR}

**********************************************************************

NTSCGEN  {NTSC COLOR BAR SIGNAL GENERATOR ROUTINE}

PROGRAMMER: DALE DUBBERT

**********************************************************************

CALLING SEQUENCE: CALL NTSCGEN
PURPOSE:

THIS ROUTINE GENERATES THE LUM AND CHR DATA ARRAYS

NOTE: ALL ARGUMENTS ARE PASSED THROUGH THE COMMON AREA NTSCCOM.FOR

******************************************************************************

SUBROUTINE NTSCGEN

INCLUDE 'NTSCCOM.FOR'

COMPLEX COLORBURST(N), ICHR(N), QCHR(N)

REAL TIME, RED, GREEN, BLUE, BLANKING, CBPULSE

******************************************************************************

GENERATE THE NTSC LUMINANCE AND EQUIVALENT LOW PASS CHROMINANCE FREQ.
DOMAIN SIGNALS

C Initialize the color burst vector

DO I = 1,N
    COLORBURST(I) = CMPLX (0.0,0.0)
END DO

C Generate the I and Q chrominance, luminance and the color burst
and blanking signals

DO I = 1,N

    Calculate the time in microseconds

    TIME = 1.0E6 * FLOAT (I-1) / (FLOAT (N) * FH)

    Find the red (r(t)), green (g(t)) and blue (b(t)) camera
    signals, the color burst signal (u(t)) and the horizontal
    sync + blanking signal (p(t)) for each iteration of time

    RED = 0.
    GREEN = 0.
    BLUE = 0.
    BLANKING = 0.
    CBPULSE = 0.

    IF ((11.0  LE. TIME .AND. 26.0  GT. TIME) .OR.
    & (41.0  LE. TIME .AND. 56.0  GT. TIME))
    & RED = 0.01 * PERSAT

    IF ((11.0  LE. TIME .AND. 18.5  GT. TIME) .OR.
    & (26.0  LE. TIME .AND. 33.5  GT. TIME) .OR.
    & (41.0  LE. TIME .AND. 48.5  GT. TIME) .OR.
    & (56.0  LE. TIME .AND. 63.6  GT. TIME))
    & BLUE = 0.01 * PERSAT

    IF (11.0  LE. TIME .AND. 41.0  GT. TIME)
    & GREEN = 0.01 * PERSAT
IF ((0.0 .LE. TIME .AND. 1.5 .GT. TIME) .OR. (6.0 .LE. TIME .AND. 11.0 .GT. TIME))
& BLANKING = -0.083
&
IF (1.5 .LE. TIME .AND. 6.0 .GT. TIME)
& BLANKING = -0.083 - 0.33
&
IF (7.0 .LE. TIME .AND. 9.5 .GT. TIME)
& CBPULSE = 0.157

C Implement the transformation matrix [A]
C add the blanking signal (p(t)) to the luminance signal (y(t))

ICHRR (I) = CMPLX (0.6 * RED - 0.377 * GREEN - 0.221 * BLUE, 0.0)
&
QCHR (I) = CMPLX (-0.191 * RED + 0.558 * GREEN - 0.368 * BLUE, 0.0)
&
LUM (I) = CMPLX (0.299 * RED + 0.587 * GREEN + 0.114 * BLUE + BLANKING, 0.0)

C CALCULATE THE COLOR BURST SIGNAL (u(t))
COLORBURST (I) = CMPLX (CBPULSE, 0.0)

END DO

C DFT the I and Q chrominance signals
CALL DFT (ICHRR,N,0)
CALL DFT (QCHR,N,0)

C Filter the I (Hv(w)) and Q (Hq(w)) chrominance signals
CALL BWLPF (ICHRR,N,IPOL,1.0E6,ICUT,FH,0.0)
CALL BWLPF (QCHR,N,QPOL,1.0E6, QCUT,FH,0.0)

C DFT the color burst signal
CALL DFT (COLORBURST,N,0)

C Filter the color burst signal using the same filter parameters as those specified for the luminance signal (Hv(w))
CALL BWLPF (COLORBURST,N,YPOL,1.0E6, YCUT,FH,0.0)

C Compute the equivalent low pass chrominance + color burst DFT coefficients
DO I = 1,N
    CHR (I) = ICHR (I) - CMPLX (0.0, 1.0)
& * (QCHR (I) - COLORBURST (I))
END DO

C Low pass filter the complex envelope of the chrominance signal (Hv(w+wc)) to obtain the equivalent low pass chrominance coefficients (c(w))
CALL BWLPF (CHR,N,YPOL,1.0E6 * YCUT,FH,FC)

C DFT the luminance signal (y(t) + p(t))
CALL DFT (LUM,N,0)

C Filter the luminance signal (Hv(w)) to obtain the luminance
DFT coefficients (l(w))
CALL BWLPF (LUM,N,YPOL,1.0E6 * YCUT,FH,0.0)

C APPLY CCIR 405-1 FM PREEMPHASIS TO THE SIGNALS IF THE USER HAS
C REQUESTED IT

C IF (PREPLG .EQ. 'Y') THEN
CALL PREEMPH (LUM,N,FH,0.0,0)
CALL PREEMPH (CHR,N,FH,FC,0)
END IF
RETURN
END

C******************************************************************************
C
C TIMEVIDEO   (SUBROUTINE TO GENERATE THE TIME DOMAIN NTSC VIDEO
C SIGNAL)
C******************************************************************************
C
C CALLING SEQUENCE: CALL TIMEVIDEO
C PURPOSE:
C THIS ROUTINE GENERATES THE VIDEOSIGNAL ARRAY.
C NOTE: ALL ARGUMENTS ARE PASSED IN THE COMMON BLOCK NTSCCOM.FOR.
C******************************************************************************

SUBROUTINE TIMEVIDEO
INCLUDE 'NTSCCOM.FOR'
COMPLEX LUMINANCE(N),CHROMINANCE(N)
REAL TIME,REALCHR,IMAGCHR
INTEGER I2

C Take the inverse DFT of the chrominance and luminance coefficients
DO I = 1,N
   LUMINANCE(I) = LUM(I)
   CHROMINANCE(I) = CHR(I)
END DO
CALL DFT (LUMINANCE,N,1)
CALL DFT (CHROMINANCE,N,1)
DO I = 1,N2
  C Calculate the n length counter
  I2 = (I + 1) / 2
  C Calculate the time in seconds
  TIME = FLOAT (I - 1) / (FH * FLOAT (N2))
  C Calculate the real and imaginary parts of the chrominance
  signal for each iteration
  REALCHR = REAL (CHROMINANCE (I2))
  IMAGCHR = AIMAG (CHROMINANCE (I2))
  C Check for illegal arguments in the ATAN2 function
  IF (REALCHR .EQ. 0.0 .AND. IMAGCHR .EQ. 0.0) REALCHR = 1.0E-30
  C Calculate the video signal for each iteration
  VIDEOSIGNAL(I) = CABS (CHROMINANCE(I2)) * COS (WC * TIME +
                & ATAN2 (IMAGCHR,REALCHR)) + REAL (LUMINANCE(I2))
END DO
RETURN
END

C**************************************************************************
C
C FREQVIDEO  (SUBROUTINE TO CALCULATE THE LOG MAGNITUDE SPECTRA
OF THE NTSC VIDEO SIGNAL)
C
C**************************************************************************

CALLING SEQUENCE: CALL FREQVIDEO
PURPOSE:
THIS ROUTINE GENERATES THE VIDEOSPECTRA ARRAY.
NOTE: ALL ARGUMENTS ARE PASSED IN THE COMMON AREA NTSCCOM.FOR.

**************************************************************************

SUBROUTINE FREQVIDEO
INCLUDE 'NTSCCOM.FOR'
REAL RARG,HIGH,LOW,MID
INTEGER COLORS

C Initialize the spectrum to a small value
DO I = 1,N2
  VIDEOSPECTRA(I) = -1.0E20
END DO

C Generate the one sided spectra of the luminance signal
DO I = 1,N2,4
  VIDEOSPECTRA(I) = 20.0 * LOG10 (CABS (LUM ((I + 3) / 4)))
END DO

C Calculate the color subcarrier pointer
COLORSC = NINT ((4.0 * FC / FH) + 1.0)

C Generate the high side spectra for the chrominance signal
DO I = COLORSC,N2,4
  VIDEOSPECTRA(I) = 20.0 * LOG10
  & (CABS (CHR ((I - COLORSC + 4) / 4)))
END DO

C Generate the low side spectra for the chrominance signal
DO I = COLORSC,3,-4
  VIDEOSPECTRA(I) = 20.0 * LOG10
  & (CABS (CHR ((I - COLORSC) / 4 + N)))
END DO

C Normalize the spectra
CALL DATSTAT (VIDEOSPECTRA,N2,HIGH,LOW,MID)
DO I = 1,N2
  VIDEOSPECTRA(I) = VIDEOSPECTRA(I) - HIGH
END DO

C Set the noise floor
DO I = 1,N2
  IF (VIDEOSPECTRA(I) .LE. NOISEFL)
    VIDEOSPECTRA(I) = NOISEFL
END DO
RETURN
END

C**************************************************************************
C
C TIMEPHASE  (SUBROUTINE TO CALCULATE THE PLL INPUT PHASE SIGNAL IN
C THE TIME DOMAIN)

C**************************************************************************

C CALLING SEQUENCE:  CALL TIMEPHASE

C PURPOSE:

C  THIS ROUTINE GENERATES THE PHASE ARRAY.

C  NOTE:  ALL ARGUMENTS ARE PASSED THROUGH THE COMMON AREA NTSCCOM.FOR.

C**************************************************************************

SUBROUTINE TIMEPHASE

INCLUDE 'NTSCCOM.FOR'
REAL TIME
INTEGER I2

C IDFT the phase coefficients
CALL DFT (PHALUM,N,1)
CALL DFT (PHACHR,N,1)

C Calculate the phase signal from the time coefficients of the
C chrominance and luminance phase signals
C The phase signal contains 8192 data points
DO I = 1,N4
   TIME = FLOAT (I-1) / (FH * FLOAT (N4))
   I2 = (I + 3) / 4
   PHASE (I) = REAL (CEXP (CMPLX (0.0, WC * TIME)) *
                PHACHR (I2)) + REAL (PHALUM (I2))
END DO

C Filter the time domain phase signal to eliminate the harmonics
C produced by increasing the data file size from N to N4
C Set the cutoff to 8 MHz and the number of poles to 5 for the
C Butterworth low pass filter
DO I = 1,N4
   CBUFFER (I) = CMPLX (PHASE (I), 0.0)
END DO
CALL DFT (CBUFFER, N4, 0)
CALL BWLPP (CBUFFER, N4, 5, 8.0E6, FH, 0.0)
CALL DFT (CBUFFER, N4, 1)

DO I = 1,N4
   PHASE (I) = REAL (CBUFFER (I))
END DO

C DFT the phase coefficients
CALL DFT (PHALUM,N,0)
CALL DFT (PHACHR,N,0)

RETURN
END

*****************************************************************************

C FMSPEC
   (SUBROUTINE TO CALCULATE THE FM SPECTRA FROM THE TIME
   DOMAIN PHASE SIGNAL)

*****************************************************************************

CALL SEQUENCE: CALL FMSPEC

PURPOSE:

   THIS ROUTINE GENERATES THE FMSPECTRA ARRAY.

NOTE: ALL ARGUMENTS ARE PASSED THROUGH THE COMMON AREA NTSCCOM.FOR.

*****************************************************************************
SUBROUTINE FMSPEC

INCLUDE 'NTSCCOM.FOR'
REAL HIGH,LOW,MID

C Compute the FM spectra from the phase signal
C The FM spectra contains 8192 data points
DO I = 1,N4
  CBUFFER(I) = CEXP (CMPLX (0.0, PHASE(I)))
END DO

CALL DFT (CBUFFER,N4,0)

DO I = 1,N4
  FMSPECTRA (I) = 20 * LOG10 (CABS (CBUFFER(I)))
END DO

C Shift the carrier frequency to the center of the sequence
DO I = 1,N4 / 2
  RBUFFER(I) = FMSPECTRA(I + N4 / 2)
  FMSPECTRA(I + N4 / 2) = FMSPECTRA(I)
  FMSPECTRA(I) = RBUFFER(I)
END DO

C Normalize the data
CALL DATSTAT (FMSPECTRA,N4,HIGH,LOW,MID)
DO I = 1,N4
  FMSPECTRA(I) = FMSPECTRA(I) - HIGH
END DO

C Set the noise floor
DO I = 1,N4
  IF (FMSPECTRA(I) <= NOISEFL)
    FMSPECTRA(I) = NOISEFL
  END DO

RETURN
END

******************************************************************************
C
*C PLLDEMOD (SUBROUTINE TO CALCULATE THE TIME DOMAIN PLL DEMODULATED SIGNAL)*
C******************************************************************************

C CALLING SEQUENCE: CALL PLLDEMOD

C PURPOSE:

THIS ROUTINE GENERATES THE DEMOD ARRAY

NOTE: ALL ARGUMENTS ARE PASSED THROUGH THE COMMON AREA NTSCCOM.FOR.

******************************************************************************
C
SUBROUTINE PLLDEMOD
INCLUDE 'NTSCCOM.FOR'
REAL W,TIM,REALCHR,IMAGCHR
CHARACTER*1 RESPONSE

C Filter the error coefficients with the loop filter transfer function
DO I = 1,N
   DEMLUM(I) = ERRLUM(I)
   DEMCHR(I) = ERRCHR(I)
END DO

DO I = 1,N/2
   W = FLOAT (I - 1) * WH
   DEMLUM(I) = DEMLUM(I) * CMPLX(1.0, T2 * 1.0E-9 * W) / CMPLX(1.0, T1 * 1.0E-9 * W)
   W = -FLOAT (I) * WH
   DEMLUM(N+1-I) = DEMLUM(N+1-I)*CMPLX(1.0,T2*1.0E-9*W) / CMPLX(1.0, T1 * 1.0E-9 * W)
   W = FLOAT (I - 1) * WH + WC
   DEMCHR(I) = DEMCHR(I) * CMPLX(1.0, T2 * 1.0E-9 * W) / CMPLX(1.0, T1 * 1.0E-9 * W)
   W = -FLOAT (I) * WH + WC
   DEMCHR(N+1-I) = DEMCHR(N+1-I)*CMPLX(1.0,T2*1.0E-9*W) / CMPLX(1.0, T1 * 1.0E-9 * W)
END DO

C Ask for deemphasis
TYPE11
111 FORMAT (/,'APPLY DEEMPHASIS? (Y/N) ---> ',A)
READ (6,112) RESPONSE
112 FORMAT (A1)

IF (RESPONSE .EQ. 'Y' .OR. RESPONSE .EQ. 'y') THEN
   TYPE*,'APPLYING DEEMPHASIS'
   CALL PREEMPH (DEMLUM,N,FH,0.0,1)
   CALL PREEMPH (DEMCHR,N,FH,FC,1)
END IF

C Calculate the time domain demodulated signal
CALL DFT (DEMLUM,N,1)
CALL DFT (DEMCHR,N,1)

DO I = 1,N2
   TIME = FLOAT(I-1) / (FLOAT(N2) * FH)
   I2 = (I + 1) / 2
   REALCHR = REAL (DEMCHR(I2))
   IMAGCHR = AIMAG (DEMCHR(I2))
   IF (REALCHR.EQ.0.0.0.AND.IMAGCHR.EQ.0.0) REALCHR = 1.0E-30
   DEMOD(I) = CAB(S (DEMCHR(I2)) * COS (WC * TIME +
   & ATAN2 (IMAGCHR,REALCHR)) + REAL (DEMLUM(I2))
END DO
CALL DFT (DELMUM,N,0)
CALL DFT (DEMCRR,N,0)

RETURN
END

*****************************************************************************
VAX-11 FORTRAN SOURCE FILENAME: NTSCFIL.FOR
DEPARTMENT OF ELECTRICAL ENGINEERING KANSAS STATE UNIVERSITY
REVISION DATE PROGRAMMER(S)
---------- ---- ------------------
4.0 5/31/86 DALE DUBERT

PROJECT: MS REPORT: RMS PHASE ERROR FOR A SECOND ORDER PHASE LOCKED
LOOP EMPLOYED AS A SATELLITE FM VIDEO DEMODULATOR

*****************************************************************************
RMS PHASE ERROR SIMULATION FREQUENCY DOMAIN FILTER SUBROUTINES:
DFT (DISCRETE FOURIER TRANSFORM)
BWLPF (BUTTERWORTH LOW PASS FILTER)
PREEMPH (FM PREEMPHASIS / DEEMPHASIS)
INTEG (INTEGRATION)
ERRFUNC (LINEAR PLL ERROR TRANSFER FUNCTION)

*****************************************************************************
DFT (DISCRETE FOURIER TRANSFORM SUBROUTINE)
PROGRAMMER: STEPHEN A. DYER

*****************************************************************************
CALLING SEQUENCE:

CALL DFT (X,NOPTS,INV)

PURPOSE:
This routine implements the FFT algorithm to compute
the DFT coefficients of a complex-valued data sequence
of NOPTS points, where NOPTS is an integer power of 2.

ARGUMENT(S) REQUIRED FROM THE CALLING ROUTINE:
X = complex-valued data/transform sequence
NOPTS = number of data points in X. NOPTS must
be an integer power of 2.
INV = flag for forward/inverse transform
INV = 0: forward DFT
INV = 1: inverse DFT

ARGUMENT(S) SUPPLIED TO THE CALLING ROUTINE:

X = complex-valued data/transform sequence resulting from the transform operation.

*************************************************************************

NOTE 1: This routine was adapted from a FORTRAN subroutine presented on page 79 of Ahmed and Rao, Orthogonal Transforms for Digital Signal Processing, Springer-Verlag, 1975. For a derivation of the algorithm, see:


*************************************************************************

SUBROUTINE DFT (X,NOPTS,INV)

IMPLICIT NONE
INTEGER I,INDEX,INDEX2,INDEX3,INV,ITER,J,J1,J2,
1 K,N1,N2,NOPTS,NPAR,NPAR2,REM
REAL ARG,PN,SIGN,WPOWER
COMPLEX CMPLX,T,W,X(*)

* Calculate the number of iterations.
ITER = 0
REM = NOPTS/2
DO WHILE (REM.NE.0)
  REM = REM/2
  ITER = ITER + 1
ENDDO

SIGN = -1.0
IF (INV.NE.0) SIGN = 1.0

NPAR2 = NOPTS
DO INDEX = 1,ITER
  * Computation for each iteration.
  * NPAR is the number of points in a partition
  * NPAR2 is NPAR/2

  NPAR = NPAR2
  NPAR2 = NPAR/2
  WPOWER = 3.1415926/FLOAT(NPAR2)
  DO INDEX2 = 1,NPAR2
    * Calculate the multiplier.
    ARG = FLOAT(INDEX2 - 1)*WPOWER
    W = CMPLX(COS(ARG),SIGN*SIN(ARG))
    DO INDEX3 = NPAR,NOPTS,NPAR
      * Computation for each partition.
J1 = INDEX3 - NPAR + INDEX2
J2 = J1 + NPAR2
T = X(J1) - X(J2)
X(J1) = X(J1) + X(J2)
X(J2) = T * W

ENDDO
ENDDO

* Unscramble the bit-reversed DFT coefficients.
N2 = NOPTS/2
J = 1
DO I = 1,NOPTS - 1
   IF (I.LT.J) THEN
      T = X(J)
      X(J) = X(I)
      X(I) = T
   ENDIF
   K = N2
   DO WHILE (K.LT.J)
      J = J - K
      K = K/2
   ENDDO
   J = J + K
ENDDO

IF (INV.EQ.0) THEN
   FN = 1.0/FLOAT(NOPTS)
   DO I = 1,NOPTS
      X(I) = X(I) * FN
   ENDDO
ENDIF
RETURN
END

*******************************************************************************

C BWLPF  (SUBROUTINE TO BUTTERWORTH LOW PASS FILTER A COMPLEX DATA
C ARRAY)

C PROGRAMMER: DALE DUBBERT

*******************************************************************************

C CALLING SEQUENCE: CALL BWLPF(ARRAY,N,POL,F3,FH,FC)
C PARAMETERS SUPPLIED TO THE SUBROUTINE:
C   ARRAY  The complex data array to be filtered
C   N      The number of data points in the array
POL  The number of poles in the desired transfer
function (POL=1,2,3,4 or 5). If POL = 0 then no
filtering is done on the data.
F3   The 3dB cutoff frequency in Hz
FH   The frequency increment between the data points
     in Hz
FC   Is the frequency offset (in Hz) for the filtering
     of a complex envelope with a carrier freq. of FC

ARGUMENTS RETURNED TO THE CALLING ROUTINE:
ARRAY  The complex filtered data array

FUNCTIONS CALLED BY THIS ROUTINE:
BUTTER  Function to compute the complex butterworth coefficient
         given a'normalized frequency and the number of poles for
         the filter

SUBROUTINE BWLPF(ARRAY,N,POL,F3,FH,FC)

INTEGER   N,   POL,   ITER
REAL       F3,   FH,   FC
COMPLEX    ARRAY(*),   BUTTER

C  If POL = 0 then return to the calling routine (no filtering)
IF(POL.EQ.0) GOTO 100

C  Filter the complex data array
DO ITER=1,N/2
   ARRAY(ITER)=ARRAY(ITER)*BUTTER((FLOAT(ITER-1)*FH+FC)
&        /F3,POL)
   ARRAY(N+1-ITER)=ARRAY(N+1-ITER)*BUTTER(((0.-FLOAT(ITER))
&        *FH+FC)/F3,POL)
END DO

C  Set the nyquist coefficient to zero
ARRAY(N/2+1)=CMPLX(0.,0.)

100  RETURN
END

C******************************************************************************
C BUTTER  (FUNCTION TO CALCULATE THE DISCRETE BUTTERWORTH FILTER
         COEFFICIENT FOR A GIVEN NORMALIZE FREQUENCY)
C******************************************************************************
C
PROGRAMMER: DALE DUBBERT
C******************************************************************************
CALLING SEQUENCE: BUTTER(NFREQ, POL)

ARGUMENTS SUPPLIED TO THE FUNCTION:

NFREQ  The ratio of the frequency of interest to
       the filter 3dB cutoff frequency.

POL    The number of filter poles. (POL=1,2,3,4 or 5)

ARGUMENTS RETURNED TO THE CALLING ROUTINE:

BUTTER  The complex butterworth filter coefficient

FUNCTION BUTTER(NFREQ, POL)

INTEGER     ITER, POL
REAL         NFREQ
COMPLEX      BUTTER

Compute the butterworth coefficient corresponding to the number
of poles
GOTO (1,2,3,4,5), POL
1    BUTTER=CMPLX(1.0,NFREQ)
   GOTO 9
2    BUTTER=CMPLX((1.0-NFREQ)**2,1.41421*NFREQ)
   GOTO 9
3    BUTTER=CMPLX(1.0-2.0*NFREQ**2,2.0*NFREQ-NFREQ**3)
   GOTO 9
4    BUTTER=CMPLX(NFREQ**4-3.414*NFREQ**2+1,2.6131*NFREQ-
     &       2.6131*NFREQ**3)
   GOTO 9
5    BUTTER=CMPLX(1.0-5.236*NFREQ**2+3.236*NFREQ**4,
     &       3.236*NFREQ-5.236*NFREQ**3+NFREQ**5)
9    BUTTER=(1.0,0.0)/BUTTER
RETURN
END

------------------------------------------------------------------------

PREEMPH     (SUBROUTINE TO APPLY FM PREEMPHASIS OR DEEMPHASIS TO A
            COMPLEX DATA ARRAY AS PER CCIR RECOMMENDATION 405-1)

PROGRAMMER: DALE DUBBERT

------------------------------------------------------------------------

CALLING SEQUENCE: CALL PREEMPH(ARRAY,N,FH,FC,PRE_DE)

ARGUMENTS SUPPLIED TO THE SUBROUTINE:

ARRAY     The complex data array

N         The size of array
FH  The frequency increment between the data points in the array
     in Hz
FC  The carrier frequency in Hz for filtering an equivalent low
     pass signal
PRE_DE  The preemphasis/deemphasis specifier

     PRE_DE=0  (PRE-EMPHASIS)
     PRE_DE=1  (DE-EMPHASIS)
     PRE_DE=-1  (NO FILTERING)

ARGUMENTS RETURNED TO THE CALLING ROUTINE:

ARRAY  The complex filtered data array

ROUTINES CALLED BY THIS ROUTINE:

PRE  Function to calculate the discrete preemphasis coefficient
      for a given frequency
DE  Function to calculate the discrete deemphasis coefficient
      for a given frequency

******************************************************************************

SUBROUTINE PREEMPH(ARRAY,N,FH,FC,PRE_DE)
INTEGER      N,   ITER,   PRE_DE
REAL         FH,   FC
COMPLEX      ARRAY(*), PRE,   DE

C  Return to the calling routine if no filtering is requested
IF (PRE_DE .EQ. -1) GOTO 100
C
C  Apply preemphasis if requested
IF(PRE_DE.EQ.0)THEN
   DO ITER=1,N/2
      ARRAY(ITER)=ARRAY(ITER)*PRE((FLOAT(ITER-1)*FH+FC))
      ARRAY(N+1-ITER)=ARRAY(N+1-ITER)*PRE((0.-FLOAT(ITER))*
         FH+FC)
   END DO
ELSE
C
C  Apply deemphasis if requested
   DO ITER=1,N/2
      ARRAY(ITER)=ARRAY(ITER)*DE((FLOAT(ITER-1)*FH+FC))
      ARRAY(N+1-ITER)=ARRAY(N+1-ITER)*DE((0.-FLOAT(ITER))*
         FH+FC)
   END DO
END IF
100  RETURN
END

******************************************************************************

C
PRE  (FUNCTION TO CALCULATE THE DISCRETE PREEMPHASIS COEFFICIENTS FOR A GIVEN FREQUENCY)

PROGRAMMER: DALE DUBBERT

ARGUMENTS SUPPLIED TO THE FUNCTION:
W  The frequency in Hz

ARGUMENTS RETURNED TO THE CALLING ROUTINE:
PRE  The complex preemphasis coefficient

FUNCTION PRE(W)
REAL  W,  R,  RL,  R3,  C,  L
COMPLEX  Z1,  Z2,  PRE

C  Convert the frequency to radians/second
W=2.0*3.141593*W

C  Check for W = 0.0
IF (W .EQ. 0.0) W = 1.0E-30

C  Set the component values for CCIR recommendation 405-1 preemphasis
R=75.
RL=275.8
R3=20.4
L=17.35E-6
C=3085.3E-12

C  Calculate the preemphasis coefficient
Z1=RL/CMPLX(1.,W*C*R1)
Z2=CMPLX(R3,W*L)

RETURN
END

DE  (FUNCTION TO CALCULATE THE DISCRETE DEEMPHASIS COEFFICIENT FOR A GIVEN FREQUENCY)

PROGRAMMER: DALE DUBBERT

ARGUMENTS SUPPLIED TO THE FUNCTION:
W  The frequency in Hz
C ARGUMENTS RETURNED TO THE CALLING ROUTINE:

DE       The complex deemphasis coefficient

FUNCTION DE(W)

REAL W, R, R1, R3, C, L
COMPLEX Z1, Z2, DE

C Convert the frequency to radians/second
W = 2.0 * 3.141593 * W

C Check for W = 0.0
IF (W .EQ. 0.0) W = 1.0E-30

C Set the component values for CCIR recommendation 405-1 deemphasis
R=75.
R1=275.8
R3=20.4
L=50.16E-6
C=8917.E-12

C Calculate the deemphasis coefficient
Z1=R1/CMPLX(1.,-R1/(W*L))
Z2=CMPLX(R3,-1.0/(W*C))

RETURN
END

C*****************************************************************************

INTEG   (SUBROUTINE TO INTEGRATE A COMPLEX DATA ARRAY OF DFT COEFFICIENTS)

PROGRAMMER: DALE DUBBERT

C*****************************************************************************

CALLING SEQUENCE: CALL INTEG(ARRAY,N,FH,FC)

ARGUMENTS SUPPLIED TO THE SUBROUTINE:

ARRAY    The complex data array
N         The array size
FH        The frequency increment between the data points in Hz
FC        The carrier frequency in Hz for integrating an equivalent low pass data array

ARGUMENTS RETURNED TO THE CALLING ROUTINE:

C ARRAY    The complex integrated array
SUBROUTINE INTEG(ARRAY,N,FH,FC)

INTEGER N, ITER
REAL FH, FC, WH, WC, W
COMPLEX ARRAY(*)

Calculate the frequencies in radians/second
WH=2.0 * 3.141593 * FH
WC=2.0 * 3.141593 * FC

Integrate the DFT coefficients
DO ITER=1,N/2
   W = FLOAT(ITER-1)*WH+WC
   IF (W .NE. 0.0) THEN
      ARRAY(ITER)=ARRAY(ITER)/CMPLX(0.,W)
   END IF
   W = -FLOAT(ITER)*WH+WC
   IF (W .NE. 0.0) THEN
      ARRAY(N+1-ITER)=ARRAY(N+1-ITER)/CMPLX(0.,W)
   END IF
END DO
RETURN
END

ERRFUNC (SUBROUTINE TO IMPLEMENT THE ERROR TRANSFER FUNCTION
OF A SECOND ORDER LINEAR PHASE LOCKED LOOP)

PROGRAMMER: DALE DUBBERT

CALLING SEQUENCE: CALL ERRFUNC (ARRAY,N,FH,FC,DAMP,FN,T1)

ARGUMENTS SUPPLIED TO THE SUBROUTINE:

ARRAY The complex data array representing the PLL phase input DFT coefficients
N The size of the complex array
FH The frequency increment between the data points in Hz
FC The carrier frequency in Hz for filtering an equivalent low
pass signal

DAMP The damping coefficient of the 2nd order PLL linear model
FN The natural frequency of the PLL model in MHz
T1 The pole time constant of the loop filter in nanoseconds

ARGUMENTS RETURNED TO THE CALLING ROUTINE:

ARRAY The DFT coefficients representing the PLL phase error

******************************************************************************

SUBROUTINE ERRFUNC (ARRAY,N,PH,FC,DAMP,FN,T1)

COMPLEX ARRAY(*)
INTEGER N, I
REAL PH, FC, DAMP, FN, T1
REAL WH, WC, WN, T, W

C Convert the external parameters to their internal values
WH = 2.0 * 3.141593 * PH
WC = 2.0 * 3.141593 * FC
WN = 2.0 * 3.141593 * FN * 1.0E6
T = T1 * 1.0E-9

C Implement the error function on the complex array
DO I = 1,N/2
  W = FLOAT(I - 1) * WH + WC
  ARRAY(I) = ARRAY(I) * CMPLX (-W**2, W / T) / 
              CMPLX(WN**2 - W**2, 2.0 * DAMP * WN * W)
  W = -FLOAT(I) * WH + WC
  ARRAY(N+1-I) = ARRAY(N+1-I) * CMPLX (-W**2, W / T) / 
                 CMPLX(WN**2 - W**2, 2.0 * DAMP * WN * W)
END DO
RETURN
END

******************************************************************************

VAX-11 FORTRAN SOURCE FILENAME: NTSCAUX.FOR
DEPARTMENT OF ELECTRICAL ENGINEERING KANSAS STATE UNIVERSITY
REVISION DATE PROGRAMMER(S)
-------- ------ ---------------
4.0 4/26/86 DALE DUBBERT
PROJECT: MS REPORT; RMS PHASE ERROR OF A SECOND ORDER PHASE
LOCKED LOOP EMPLOYED AS A SATELLITE FM VIDEO DEMODULATOR.

*********************************************************************************************************************************************

AUXILIARY SUBROUTINES FOR THE NTSC PHASE ERROR SIMULATION:

DATA STORAGE AND RETRIEVAL:

FIGET
FSTORE

MENU GENERATORS:

MENGET
MENGEN
MENMXT1
MENPMPLL
MENMXT2

PARAMETER STORAGE AND RETRIEVAL:

GETPARA
SHOWPARA
STOREPARA

MISC:

DATSTAT
PHAERROR
WAIT_RET

*********************************************************************************************************************************************

*********************************************************************************************************************************************

DATSTAT     (SUBROUTINE TO COMPUTE THE MAXIMUM, MINIMUM AND THE AVERAGE VALUE OF A REAL DATA ARRAY)

*********************************************************************************************************************************************

CALLING SEQUENCE: CALL DATSTAT (ARRAY,N,MAX,MIN,AVE)

ARGUMENTS SUPPLIED TO THE SUBROUTINE:

ARRAY     The real data array
N         The length of the data array

ARGUMENTS RETURNED TO THE CALLING ROUTINE:

MAX       The maximum value in the array
MIN       The minimum value in the array
AVE       The average value of the array

*********************************************************************************************************************************************
SUBROUTINE DATSTAT(ARRAY,N,MAX,MIN,AVE)

INTEGER N, ITER
REAL ARRAY(*), MAX, MIN, AVE

MAX=ARRAY(1)
MIN=ARRAY(1)
AVE=ARRAY(1)

DO ITER=2,N
   IF(ARRAY(ITER) .GT. MAX) MAX=ARRAY(ITER)
   IF(ARRAY(ITER) .LT. MIN) MIN=ARRAY(ITER)
   AVE=AVE+ARRAY(ITER)
END DO

AVE=AVE/FLOAT(N)

RETURN
END

C******************************************************************************
C PHAERROR  (SUBROUTINE TO CALCULATE THE RMS PHASE ERROR OF THE LINEAR PLL
GIVEN THE CHROMINANCE AND LUMINANCE PHASE ERROR DFT
COEFFICIENTS)
C******************************************************************************
C CALLING SEQUENCE:  CALL PHAERROR
C NOTE: ALL ARGUMENTS ARE PASSED THROUGH THE COMMON AREA NTSCCOM.FOR
C******************************************************************************

SUBROUTINE PHAERROR

INCLUDE 'NTSCCOM.FOR'

C IDFT the error coefficients
CALL DFT (ERRLUM,N,1)
CALL DFT (ERRCHR,N,1)

C Compute the phase error from the ERRLUM and ERRCHR time coefficients
RMSERROR = 0.0
DO I = 1,N
   RMSERROR = RMSERROR + 0.5 * CABS(ERRCHR(I))**2 + CABS(ERRLUM(I))**2
&
END DO
RMSERROR = SQRT(RMSERROR/FLOAT(N))

C DFT the error coefficients
CALL DFT (ERRLUM,N,0)
CALL DFT (ERRCHR,N,0)

RETURN
END
FISTORE (SUBROUTINE TO STORE A DATA ARRAY TO AN EXTERNAL FILE)

CALLING SEQUENCE: CALL FISTORE(DEFAULT, NUM, TYPE, VECTOR)

PARAMETERS SUPPLIED TO THE SUBROUTINE:

DEFAULT The default file name which is used if the user specifies no file name (character)
NUM The number of data elements in the array (integer)
TYPE The array type ('REAL' or 'COMPLEX')
VECTOR The internal data array

PARAMETERS RETURNED TO THE CALLING ROUTINE:

NONE

SUBROUTINES CALLED BY THIS ROUTINE:

SGOPEN: Routine to open an external file for data storage or retrieval
SGTRAN: Routine to transfer data between an external file and an internal array

SUBROUTINE FISTORE(DEFAULT, NUM, TYPE, VECTOR)

INTEGER NUM
CHARACTER*15 FINAME
CHARACTER*(*) DEFAULT, TYPE
REAL VECTOR(*)

Prompt for entry of and read the data file name

TYPE*, 'ENTER THE FILE NAME (15 CHARACTERS MAX)'
TYPE111, DEFAULT
FORMAT (T2,'THE DEFAULT IS ',A, ' ----> ',$)
READ (5,5) FINAME
5 FORMAT (A15)

Use default file name if no name is specified by the user

IF(FINAME.EQ. ' ') FINAME=DEFAULT

Check for valid data type. If type not valid then print an error message and return to the calling routine.

IF('TYPE.EQ. 'COMPLEX' .OR. TYPE.EQ. 'REAL') GOTO 50
TYPE*, '*****INVALID DATA TYPE***'
TYPE*, '*****NO FILES READ*****'
GOTO 40

C
Open the external data file and store the data array

50 CALL SGOPEN(1, 'WRITE', 'NOPROMPT', FINAME, TYPE, NUM)
CALL SGTRAN(1, 'WRITE', TYPE, VECTOR, NUM)

C
Display the file name, type and the number of data elements

TYPE10, FINAME, TYPE, NUM

RETURN
END

C*****************************************************************************
C  FIGET    (SUBROUTINE TO RETRIEVE A DATA ARRAY FROM AN EXTERNAL FILE)
C*****************************************************************************

CALLING SEQUENCE: CALL FIGET (DEFAULT, NUM, TYPE, VECTOR)

PARAMETERS SUPPLIED TO THE SUBROUTINE:

    DEFAULT        The default file name which is used if the user
                   specifies no file name (character)

PARAMETERS RETURNED TO THE CALLING ROUTINE:

    NUM            The number of data elements in the array (integer)
    TYPE           The array type ('REAL' or 'COMPLEX')
    VECTOR        The internal data array

SUBROUTINES CALLED BY THIS ROUTINE:

    SGOPEN: Routine to open an external file for data storage
            or retrieval
    SGTRAN: Routine to transfer data between an external file
            and an internal array

SUBROUTINE FIGET (DEFAULT, NUM, TYPE, VECTOR)

INTEGER NUM
CHARACTER*15 FINAME
CHARACTER(*) DEFAULT, TYPE
REAL VECTOR(*)

C
Prompt for entry of and read the data file name
TYPE*, 'ENTER THE FILE NAME (15 CHARACTERS MAX)'
TYPEII, DEFAULT
111 FORMAT (T2, 'THE DEFAULT IS ',A,' ----> ',$,)
READ (6,5) FINAME
5 FORMAT (A15)
C Use default file name if no name is specified by the user
IF(FINAME.EQ.'
           ') FINAME = DEFAULT
C Open and read the external data file
CALL SGOPEN(1,'READ','NOPROMPT',FINAME,TYPE,NUM)
CALL SGTRAN(1,'READ',TYPE,VECTOR,NUM)
C Display the file name, type and number of data elements read
TYPE10,FINAME,TYPE,NUM
10 FORMAT (A1,' THE ',A,' FILE IS ',A,' AND HAS ',I5,', ELEMENTS')
RETURN
END

C*******************************************************************************
C MENGET       (SUBROUTINE TO GENERATE THE MENU FOR SELECTION OF THE SOURCE
C OF THE DFT COEFFICIENTS FOR THE PHASE ERROR SIMULATION)
C*******************************************************************************
C CALLING SEQUENCE: CALL MENGET
C*******************************************************************************
C SUBROUTINE MENGET(CHOICE)
INTEGER CHOICE
INTEGER*4 STATUS
C Output the menu to the terminal
STATUS = LIBSERASE_PAGE(1,1)
TYPE*, 'MENU: GET NTSC COEFFICIENTS'
TYPE*, '1. GENERATE THE NTSC DFT COEFFICIENTS'
TYPE*, '2. RECALL STORED DFT COEFFICIENTS'
TYPE*, ' ' 100 TYPEII
111 FORMAT (T2, 'ENTER SELECTION ----> ',$,)
C Read the menu selection and return to the calling routine if
C the menu selection is valid
READ (6,1) CHOICE
1 FORMAT (I2)
IF (CHOICE.LT.1.0R. CHOICE.GT.2) THEN
    TYPE*, 'INVALID SELECTION, TRY AGAIN'
GOTO 100
END IF
RETURN
END

C******************************************************************************
C
C MENGEN  (SUBROUTINE TO DISPLAY AND EDIT THE NTSC COLOR BAR GENERATOR
C AND PREEMPHASIS PARAMETERS)
C******************************************************************************
C
CALLING SEQUENCE: CALL MENGEN
C
NOTE: ALL PARAMETERS ARE PASSED THROUGH THE COMMON AREA NTSCCOM.FOR
C******************************************************************************

SUBROUTINE MENGEN

INCLUDE 'NTSCCOM.FOR'

INTEGER PENTRY, CHOICE
REAL RENTRY
CHARACTER*1 CHENTRY

C Display the ntsc generator parameter menu and prompt the user for a selection

100 STATUS = LIBSERASE_PAGE(1,1)
    TYPE*, 'MENU: NTSC SIGNAL GENERATOR PARAMETER SELECTION'
    TYPE*, '1.'
    TYPE1, 1. # OF POLES FOR Y LUMINANCE SIGNAL = ',YPOL
    TYPE1, 2. # OF POLES FOR Q CHROMINANCE SIGNAL = ',QPOL
    TYPE1, 3. # OF POLES FOR I CHROMINANCE SIGNAL = ',IPOL
    FORMAT(A,3X,X1)
    TYPE2, 4. BANDWIDTH FOR Y LUMINANCE FILTER = ',YCYUT, 'MHZ'
    TYPE2, 5. BANDWIDTH FOR Q CHROMINANCE FILTER = ',QCYUT, 'MHZ'
    TYPE2, 6. BANDWIDTH FOR I CHROMINANCE FILTER = ',ICYUT, 'MHZ'
    FORMAT(A,1X,F7.3,1X,A)
    TYPE3, 7. CCIR 405-1 PREEMPHASIS? (Y/N) = ',PREFLG
    FORMAT(A,3X,A)
    TYPE4, 8. PERCENTAGE COLOR SATURATION = ',PERSAT, %'
    FORMAT(A,1X,F7.3,1X,A)
    TYPE*, '111
111 FORMAT (T2,'ENTER SELECTION, <RETURN> TO CONTINUE -----> ',S)

200 READ (6,10) CHOICE
10 FORMAT(II)
C Check for out of range selection
C If out of range then read keyboard again
IF (CHOICE.LT.0.OR.CHOICE.GT.8) THEN
  TYPE*, 'INVALID SELECTION, TRY AGAIN'
  GOTO 200
END IF

MENU SELECTIONS 1, 2 OR 3: CHANGE A POLE SPECIFICATION

IF (CHOICE.GE.1.AND.CHOICE.LE.3) THEN
  TYPE*, 'ENTER NUMBER OF POLES (1 TO 5, 0 = NO FILTERING)'
  FORMAT (T2, '----> ', $)
  READ (6,20) PENTRY
  FORMAT (I1)
  C
  Check for out of range pole specification
  If out of range, then read keyboard again
  IF (PENTRY.LT.0.OR.PENTRY.GT.5) THEN
    TYPE*, 'OUT OF RANGE, VALID ENTRY = 0 TO 5'
    GOTO 250
  END IF
  C
  Set the proper pole parameter to the value entered
  IF (CHOICE.EQ.1) YPOL = PENTRY
  IF (CHOICE.EQ.2) QPOL = PENTRY
  IF (CHOICE.EQ.3) IPOL = PENTRY
  C
  Return to the parameter menu display
  GOTO 100
END IF

MENU SELECTIONS 4, 5 OR 6: CHANGE A FILTER BANDWIDTH SPECIFICATION

IF (CHOICE.GE.4.AND.CHOICE.LE.6) THEN
  TYPE113
  FORMAT (T7, 'ENTER THE BANDWIDTH IN MHz ----> ', $)
  READ (6,*), RENTRY
  RENTRY = ABS(REENTRY)
  C
  Set the corresponding bandwidth parameter to the entry value
  IF (CHOICE.EQ.4) YCUT = RENTRY
  IF (CHOICE.EQ.5) QCUT = RENTRY
  IF (CHOICE.EQ.6) ICUT = RENTRY
  C
  Return to the parameter menu
  GOTO 100
END IF
MENU SELECTION 7: CHANGE THE PREEMPHASIS REQUEST FLAG

IF (CHOICE.EQ.7) THEN
  TYPE114
  FORMA(T,2,'APPLY PREEMPHASIS? (Y/N) ----> ',$,)
  READ (6,40) CENTRY
  FORMAT (A)

  Check for invalid entry
  IF (CENTRY.NE.'Y'.AND.CENTRY.NE.'N').AND.CENTRY.NE.'y'
    TYPE*,' INVALID ENTRY, ENTER 'Y' OR 'N'
  GOTO 300
  END IF

  Set the preemphasis flag to its proper upper case value
  IF (CENTRY.EQ.'Y'.OR.CENTRY.EQ.'y') PREFLG = 'Y'
  IF (CENTRY.EQ.'N'.OR.CENTRY.EQ.'n') PREFLG = 'N'

  Return to the parameter menu
  GOTO 100
END IF

MENU SELECTION 8: CHANGE THE PERCENTAGE COLOR SATURATION

IF (CHOICE.EQ.8) THEN
  TYPE*, 'ENTER THE PERCENTAGE SATURATION (0. TO 200.)'
  TYPE115
  FORMA(T,2,'----> ',$,)
  READ (6,*) RENTRY
  RENTRY = ABS(RENTY)

  Check for out of range entry
  IF entry is out of range then read the keyboard again
  IF (RENTY.GT.200.) THEN
    TYPE*, 'ENTRY OUT OF RANGE, TRY AGAIN'
  GOTO 350
  END IF

  Set the percentage saturation to the value entered
  PERSAT = RENTRY

  Return to the parameter menu
  GOTO 100
END IF

RETURN TO THE CALLING ROUTINE IF SELECTION = 0
IF (CHOICE.EQ.0) RETURN
END

C*****************************************************************************
C MENNX1 (SUBROUTINE TO DISPLAY THE MENU FOR THE NEXT ACTION
AFTER GENERATING THE CHROMINANCE AND LUMINANCE DFT
COEFFICIENTS)
C*****************************************************************************
C CALLING SEQUENCE: CALL MENNX1 (CHOICE)
C ARGUMENT SUPPLIED TO THE CALLING ROUTINE:
C CHOICE The integer menu selection
C*****************************************************************************

SUBROUTINE MENNX1 (CHOICE)

INTEGER CHOICE

C Display the menu
TYPE*, 'MENU: GENERATE VIDEO SIGNALS'
TYPE*, ''
TYPE*, '1. GENERATE THE TIME DOMAIN VIDEO SIGNAL'
TYPE*, '2. GENERATE THE LOG MAGNITUDE SPECTRA OF THE VIDEO SIGNAL'
TYPE*, '3. STORE THE CURRENT NTSC PARAMETERS (NTSCPAR.TXT)'
TYPE*, '4. RESTART THE PROGRAM'
TYPE*, ''
TYPE*, 'ENTER SELECTION OR PRESS <RETURN> TO CONTINUE WITH THE PHASE'
TYPE11
111 FORMAT (T2, 'ERROR SIMULATION -----> ',)$

10 READ(6,1) CHOICE
1 FORMAT(I2)

C Check for out of bounds entry
IF (CHOICE .LT. 0 .OR. CHOICE .GT. 4) THEN
   TYPE*, 'INVALID SELECTION, TRY AGAIN'
   GOTO 10
END IF

RETURN
END

C*****************************************************************************
C MENFMPIL (SUBROUTINE TO GENERATE THE MENU TO EDIT THE FM
TRANSMITTER AND PLL DEMODULATOR PARAMETERS)
C*****************************************************************************
C CALLING SEQUENCE: CALL MENFMPIL
NOTE: ALL PARAMETERS ARE PASSED THROUGH THE COMMON AREA NTSCCOM.FOR

SUBROUTINE MENFMPLL
INCLUDE 'NTSCCOM.FOR'
INTEGER CHOICE

C
DISPLAY THE MENU

STATUS = LIB$ERASE_PAGE(1,1)

TYPE*, 'MENU: FM MODULATOR (VCO) / DEMODULATOR (PLL) PARAMETERS'
TYPE*, ' VIDEO STATUS: MAXIMUM = ',MAX
TYPE1, FORMAT (A,F10.4)
TYPE1, MINIMUM = ',MIN
TYPE1, AVERAGE = ',AVE

TYPE*, ' FM MODULATOR (VCO)'
TYPE*, '-------------------'

TYPE2, 1. PEAK FREQUENCY DEVIATION = ',FMPKDEV, ' MHz'
TYPE2, FORMAT (A,F10.4,A)

TYPE2, 2. MODULATION INDEX Km = ',Km, ' MHz/volt'

TYPE*, ' FM DEMODULATOR (LINEAR PLL)'
TYPE*, '-------------------'

TYPE3, DEPENDENT PARAMETERS: NATURAL FREQ. Fn = ',FN, ' MHz'
TYPE3, FORMAT (A,F10.4,A)

TYPE4, DAMPING FACTOR = ',DAMP
TYPE4, FORMAT (A,F10.5)

TYPE*, ' P3. POLE TIME CONSTANT T1 = ',T1, ' nsec'
TYPE5, FORMAT (A,F10.5,A)

TYPE5, 4. ZERO TIME CONSTANT T2 = ',T2, ' nsec'

TYPE6, 5. OPEN LOOP GAIN KDKV = ',KDKV, ' 1/sec'
TYPE6, FORMAT (A,F10.4,A)

TYPE*, ' ENTER SELECTION OR <RETURN> TO CONTINUE WITH ' TYPE116

FORMAT (T2,'THE PHASE ERROR CALCULATION ------> ',S)

READ (6,10) CHOICE

C
IF (CHOICE .LT. 0 .OR. CHOICE .GT. 5) THEN
TYPE*, 'INVALID SELECTION, TRY AGAIN'
GOTO 20
END IF

MENU SELECTION #1: CHANGE THE PEAK FREQUENCY DEVIATION

IF (CHOICE .EQ. 1) THEN
  TYPE*, 'ENTER THE PEAK FREQUENCY DEVIATION (MHz)'
  TYPE117
  FORMAT (T2, '-----', $)
  READ (6, *) FMPKDEV
  C
  Calculate KM in MHz/volt
  KM = 2.0 * FMPKDEV / (MAX - MIN)
  GOTO 100
END IF

MENU SELECTION #2: CHANGE THE MODULATION INDEX Km

IF (CHOICE .EQ. 2) THEN
  TYPE118
  FORMAT (T2, 'ENTER Km (MHz/volt) ------', $)
  READ (6, *) KM
  C
  Calculate the peak deviation in MHz
  FMPKDEV = KM * (MAX - MIN) / 2.0
  GOTO 100
END IF

MENU SELECTION #3: CHANGE THE POLE TIME CONSTANT T1

IF (CHOICE .EQ. 3) THEN
  TYPE119
  FORMAT (T2, 'ENTER T1 (nsec) ------', $)
  READ (6, *) T1
  C
  Calculate Fm (MHz), Wn (rad/sec), and the dampening factor
  WN = SQRT (KDKV / (T1 * 1.0E-9))
  FM = WN / (2.0 * PI * 1.0E6)
  DAMP = 0.5 * (1.0 + KDKV * T2 * 1.0E-9) / SQRT (KDKV * T1 * 1.0E-9)
  GOTO 100
END IF

C***********************************************************************
C MENU SELECTION #4: CHANGE THE ZERO TIME CONSTANT T2
C***********************************************************************
IF (CHOICE .EQ. 4) THEN

120 TYPE120
FORMAT (T2, 'ENTER T2 (nsec) ---> ', $)
READ (6,*) T2

C Calculate the dampening factor
DAMP = 0.5 * (1.0 + KDKV * T2 * 1.0E-9) / SQRT (KDKV * T1
& * 1.0E-9)
GOTO 100
END IF

C***********************************************************************
C MENU SELECTION #5: CHANGE THE OPEN LOOP GAIN KDKV
C***********************************************************************
IF (CHOICE .EQ. 5) THEN

121 TYPE121
FORMAT (T2, 'ENTER KDKV (1/sec) ---> ', $)
READ (6,*) KDKV

C Calculate Fm (MHz), Wn (rad/sec), and the dampening factor
WN = SQRT (KDKV / (T1 * 1.0E-9))
FM = Wn / (2.0 * PI * 1.0E6)
DAMP = 0.5 * (1.0 + KDKV * T2 * 1.0E-9) / SQRT (KDKV * T1
& * 1.0E-9)
GOTO 100
END IF

RETURN ENDED

***********************************************************************
C MENNX2 (SUBROUTINE TO DISPLAY THE MENU FOR THE NEXT ACTION
AFTER THE RMS PHASE ERROR CALCULATION)
CALLING SEQUENCE: CALL MENNXT2 (CHOICE)
ARGUMENT RETURNED TO THE CALLING ROUTINE:
CHOICE The user menu selection (integer)

SUBROUTINE MENNXT2 (CHOICE)

INTEGER CHOICE

Display the menu
STATUS = LIB$ERASE_PAGE(1,1)

TYPE*, 'MENU: DISPLAY PHASE, FM SPECTRA AND DEMODULATED'
  TYPE*, 'SIGNAL'
  TYPE*, '
  TYPE*, '1. GENERATE THE PLL INPUT PHASE SIGNAL'
  TYPE*, '2. GENERATE THE PLL INPUT FM SPECTRA'
  TYPE*, '3. GENERATE THE PLL DEMODULATED SIGNAL'
  TYPE*, '4. SELECT NEW FM MODULATOR / PLL PARAMETERS'
  TYPE*, '5. SELECT NEW NTSC SIGNAL GENERATOR PARAMETERS'
  TYPE*, '6. STORE THE CURRENT PARAMETERS (NTSCPARA.TXT)'
  TYPE*, '7. RECALL THE STORED NTSC PARAMETERS (NTSCPARA.TXT)'
  TYPE*, '8. DISPLAY THE STORED NTSC PARAMETERS (NTSCPARA.TXT)'
  TYPE*, '122

FORMAT (T2, 'ENTER SELECTION OR <RETURN> TO QUIT ------ ', S)

READ (6, 1) CHOICE

Check for an out of range entry
IF (CHOICE .LT. 0 .OR. CHOICE .GT. 8) THEN
  TYPE*, 'INVALID SELECTION, TRY AGAIN'
GOTO 10
END IF

RETURN

END

SUBROUTINE WAIT_RET

(SUBROUTINE TO WAIT UNTIL THE USER HITS <RETURN>)

CALLING SEQUENCE: CALL WAIT_RET

SUBROUTINE WAIT_RET

CHARACTER*1 DUMMY

FORMAT (T2, 'PRESS <RETURN> TO CONTINUE', S)
READ(6,1) DUMMY
FORMT(A1)
RETURN
END

C******************************************************************************
C
C SHOWPARA (SUBROUTINE TO TYPE THE NTSC PARAMETER FILE (NTSCPARA.TXT)
C ON THE SCREEN)
C******************************************************************************
C
C CALLING SEQUENCE: CALL SHOWPARA
C******************************************************************************

SUBROUTINE SHOWPARA
CHARACTER*1 DUMMY
CHARACTER*80 LINE(22)

C Open the parameter text file
OPEN (UNIT=1,STATUS='OLD',FILE='NTSCPARA.TXT')

C Read the text in the file 'NTSCPARA.TXT'
REWIND (UNIT=1)
DO I = 1,22
   READ(1,1) LINE(I)
1   FORMAT (T1,A80)
END DO

C Clear the screen and output the text
STATUS=LIB$EREASE_PAGE(1,1)
DO I = 1,22
   TYPE2,LINE(I)
2   FORMAT (X,A80)
END DO

C Close the text file
CLOSE (UNIT = 1)

C Prompt for a <RETURN> to continue
TYPE3
FORMAT (T2,'PRESS <RETURN> TO CONTINUE',$,)
READ (6,4) DUMMY

RETURN
END
C*******************************************************************************
C GETPARA  (SUBROUTINE TO READ THE DEFAULT NTSC PARAMETERS FROM
C           THE NTSCPARA.TXT FILE)
C*******************************************************************************
C CALLING SEQUENCE:  CALL GETPARA
C NOTE:  ALL PARAMETERS ARE PASSED THROUGH THE COMMON AREA NTSCCOM.FOR
C*******************************************************************************
SUBROUTINE GETPARA

INCLUDE 'NTSCCOM.FOR'

C Open the text file
OPEN (UNIT=1,STATUS='OLD',FILE='NTSCPARA.TXT')

C Read the ntsc generator parameters
REWIND (UNIT=1)
READ(1,10) YCUT,QCUT,ICUT,YPOL,QPOL,IPOL,PREFLG,PERSAT
10 FORMAT ('''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
C NOTE: ALL PARAMETERS ARE PASSED THROUGH THE COMMON AREA NTSCCM.COM.FOR
C
SUBROUTINE STOREPARA

INCLUDE 'NTSCCM.COM.FOR'

CHARACTER*80 LINE(22)

C Open the old parameter file and read the text
OPEN (UNIT=1,STATUS='OLD',FILE='NTSCPARA.TXT')
REWIND (UNIT = 1)
DO I = 1,22
    READ(1,1) LINE(I)
END DO

C Close the old parameter file and open a new version
CLOSE (UNIT = 1)
OPEN (UNIT=1,STATUS='NEW',FILE='NTSCPARA.TXT')

C Output the text and the parameter values to the new file
DO I = 1,8
    WRITE(1,50) LINE(I)
END DO

WRITE (1,51) ' | ',YCUT,' | ',QCUR,' | ',ICUT,' | ',YPOL,' | ',QPOL,' | ',IPOL,' | ',PREFLG,' | ',PERSAT,' | '
& 50 FORMAT (T1,A)
& T38,A,T41,I1,T45,A,T48,I1,T52,A,T58,A,T66,A,T70,F7.3,T79,A)
DO I = 10,14
    WRITE(1,50) LINE(I)
END DO

WRITE (1,52) ' | ',FMPKDEV,' | ',KM,' | '
52 FORMAT (T1,A,T14,F7.3,T39,A,T54,F7.3,T79,A)
DO I = 16,20
    WRITE(1,50) LINE(I)
END DO

WRITE (1,53) ' | ',KDKV,' | ',T1,' | ',T2,' | ',FN,' | ',DAMP,' | '
& T44,F9.3,T54,A,T59,F8.5,T73,A)
WRITE(1,50) LINE (22)

C Close the file
CLOSE (UNIT = 1)

RETURN
END
# NTSC PARA.TXT

## NTSC COLOR BAR GENERATOR

<table>
<thead>
<tr>
<th>VIDEO FILTERS</th>
<th>PREEMPHASIS</th>
<th>PERCENT</th>
<th>FLAG</th>
<th>COLOR</th>
<th>SATURATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cutoff Frequency (MHz)</td>
<td># of Poles</td>
<td>y(t)</td>
<td>q(t)</td>
<td>i(t)</td>
<td>y(t)</td>
</tr>
<tr>
<td>4.2000</td>
<td>0.5000</td>
<td>1.5000</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

### FM TRANSMITTER (VCO)

<table>
<thead>
<tr>
<th>PEAK FREQUENCY DEVIATION (MHz)</th>
<th>MODULATION INDEX Km (MHz/volt)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.700</td>
<td>36.507</td>
</tr>
</tbody>
</table>

### FM DEMODULATOR (LINEAR PLL MODEL)

<table>
<thead>
<tr>
<th>KdKv (l/sec)</th>
<th>T1 (nsec)</th>
<th>T2 (nsec)</th>
<th>Fm (MHz)</th>
<th>Damping Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2310E+09</td>
<td>26.005</td>
<td>10.670</td>
<td>15.000</td>
<td>0.70682</td>
</tr>
</tbody>
</table>
Appendix 2. Program Operating Instructions
PROGRAM OPERATING INSTRUCTIONS FOR THE NTSC PLL PHASE ERROR SIMULATION

Program Description

The PLL phase error simulation program was written to perform the data generation, calculations and signal processing necessary to evaluate the performance of a second order PLL operating as an FM demodulator for standard NTSC video. The program was written in VAX-11 FORTRAN and structured primarily as a menu driven routine. Once the user understands the theory and the process for obtaining the phase error and related information, then the program operation is relatively straightforward.

Running the Simulation Program

The source code for the simulation is located in the following files:

NTSC.FOR
NTSCCOM.FOR
NTSCFIL.FOR
NTSCGEN.FOR
NTSCAUX.FOR

The text file which contains the default system parameters is NTSCPARA.TXT
The program can be compiled and linked using the following VAX/VMS commands:

```
FORTRAN NTSC,NTSCFIL,NTSCGEN,NTSCAUX
LINK NTSC,NTSCFIL,NTSCGEN,NTSCAUX
```

To execute the program, enter:

```
RUN NTSC
```

The program will then display a series of command and parameter menus which control the execution of the simulation.

**Menu Descriptions**

The following menus are displayed in the order of appearance during execution of the program. A sequential flow of the program is achieved by using the `<RETURN>` key after each menu. Certain menus, however, provide the user with the capability of accessing previous menus for situations where multiple phase error or data calculations are required.

**Generating the NTSC Test Signal**

The first menu asks the user to specify the source of the DFT coefficients for the chrominance and luminance components of the simulation test signal.
MENU: GET NTSC COEFFICIENTS

1. GENERATE THE NTSC DFT COEFFICIENTS
2. RECALL STORED DFT COEFFICIENTS

ENTER SELECTION ----> 

A selection of #1 allows the user to specify the parameters for the NTSC color bar pattern generator routine. The NTSC parameter menu is as follows:

MENU: NTSC SIGNAL GENERATOR PARAMETER SELECTION

1. # OF POLES FOR Y LUMINANCE SIGNAL = 3
2. # OF POLES FOR Q CHROMINANCE SIGNAL = 3
3. # OF POLES FOR I CHROMINANCE SIGNAL = 3
4. BANDWIDTH FOR Y LUMINANCE FILTER = 4.200 MHZ
5. BANDWIDTH FOR Q CHROMINANCE FILTER = 0.500 MHZ
6. BANDWIDTH FOR I CHROMINANCE FILTER = 1.500 MHZ
7. CCIR 405-1 PREEMPHASIS? (Y/N) = Y
8. PERCENTAGE COLOR SATURATION = 100.000 %

ENTER SELECTION, <RETURN> TO CONTINUE ---->

The initial defaults for this menu are located in the NTSCPARA.TXT file. Once the desired parameter values have been selected, the user hits <RETURN> to generate the NTSC test signal. The program then responds with:

GENERATING THE NTSC COEFFICIENTS

The generated chrominance and luminance files may be stored on disk if the user so desires. The following prompts and entries are for storage of the coefficients in the default files LUM.DAT and CHR.DAT:
STORE THE DFT COEFFICIENTS? (Y/N) ----> Y

STORE THE LUMINANCE DFT COEFFICIENTS (L(w))
ENTER THE FILE NAME (15 CHARACTERS MAX)
THE DEFAULT IS LUM.DAT ---->
THE LUM.DAT FILE IS COMPLEX AND HAS 1024 ELEMENTS

STORE THE CHROMINANCE DFT COEFFICIENTS (C-(w))
ENTER THE FILE NAME (15 CHARACTERS MAX)
THE DEFAULT IS CHR.DAT ---->
THE CHR.DAT FILE IS COMPLEX AND HAS 1024 ELEMENTS
PRESS <RETURN> TO CONTINUE

This program uses the same general procedure shown above for storage and retrieval of all data files. The user is given a default filename which can be used by pressing <RETURN> or a different filename can be entered. The program also responds with the data type and the length of the file for user reference.

Recalling the NTSC Signal Files From Disk

A selection of #2 from the GET NTSC COEFFICIENTS menu allows the user to input the complex, 1024 point data files from disk. The user is asked for the names of the chrominance and luminance files by the following prompts:

RECALL THE LUMINANCE DFT COEFFICIENTS:
ENTER THE FILE NAME (15 CHARACTERS MAX)
THE DEFAULT IS LUM.DAT ---->
THE LUM.DAT FILE IS COMPLEX AND HAS 1024 ELEMENTS

RECALL THE CHROMINANCE DFT COEFFICIENTS:
ENTER THE FILE NAME (15 CHARACTERS MAX)
THE DEFAULT IS CHR.DAT ---->
THE CHR.DAT FILE IS COMPLEX AND HAS 1024 ELEMENTS
PRESS <RETURN> TO CONTINUE
Default filenames of LUM.DAT and CHR.DAT are used in the cases shown above.

Generating the Time and Frequency Domain Video Signals

The time domain video signal $v(t)$ or the log magnitude spectrum of the video signal may be calculated and stored on disk by using the following menu:

**MENU: GENERATE VIDEO SIGNALS**

1. GENERATE THE TIME DOMAIN VIDEO SIGNAL
2. GENERATE THE LOG MAGNITUDE SPECTRA OF THE VIDEO SIGNAL
3. STORE THE CURRENT NTSC PARAMETERS (NTSCPARA.TXT)
4. RESTART THE PROGRAM

ENTER SELECTION OR PRESS <RETURN> TO CONTINUE WITH THE PHASE ERROR SIMULATION ---->

A selection of #1 or #2 generates a 2048 point real data file which may be plotted using an external graphics routine. The user is prompted for a disk filename (or default) in each case.

When generating the log spectra of the video signal (Selection #2), the user is asked to specify a noise floor for the log data:

ENTER THE NOISE FLOOR FOR THE NORMALIZED LOG SPECTRA
THE DEFAULT IS -80.0 dB ---->

The default of -80.0 dB provides satisfactory results in most cases.

Selection #3 allows the user to store the current NTSC generator parameters on the permanent disk file NTSCPARA.TXT.
Selection #4 returns the program to the GET NTSC COEFFICIENTS menu. By pressing <RETURN>, control is then transferred to the FM modulator (VCO) and demodulator (PLL) parameter selection menu.

**FM Modulator/Demodulator Parameter Selection and Phase Error Calculation**

The following menu displays and edits the parameters for the FM modulator (VCO) and the phase-locked loop (PLL) demodulator:

**MENU: FM MODULATOR (VCO) / DEMODULATOR (PLL) PARAMETERS**

**VIDEO STATUS:**
- **MAXIMUM =** 0.4298
- **MINIMUM =** -0.2825
- **AVERAGE =** 0.0464

**FM MODULATOR (VCO)**

1. **PEAK FREQUENCY DEVIATION** = 10.7000 MHz
2. **MODULATION INDEX Km** = 30.0440 MHz/volt

**FM DEMODULATOR (LINEAR PLL)**

**DEPENDENT PARAMETERS:**
- **NATURAL FREQ. Fn** = 15.0002 MHz
- **DAMPING FACTOR** = 0.70682

3. **POLE TIME CONSTANT T1** = 26.00500 nsec
4. **ZERO TIME CONSTANT T2** = 10.67000 nsec
5. **OPEN LOOP GAIN KdKv** = 0.2310E+09 1/sec

ENTER SELECTION OR <RETURN> TO CONTINUE WITH THE PHASE ERROR CALCULATION ---->

The FM modulator parameters (selections 1 and 2) are interdependent, i.e., by changing one, the other is recomputed based on the video status information provided above the menu.
The dependent PLL parameters are computed for each new selection of the independent PLL parameters (selections 3, 4 and 5).

Once the desired parameters have been selected, a <RETURN> will continue with the RMS phase error calculation. A sample output for the aforementioned default parameters is

RMS PHASE ERROR = 0.1412961 radians = 8.095671 degrees
PRESS <RETURN> TO CONTINUE.

Computing the Phase, FM Spectra and the Demodulated Test Signal

After computing the phase error, the next menu provides the user with several options.

MENU: DISPLAY PHASE, FM SPECTRA AND DEMODULATED SIGNAL

1. GENERATE THE PLL INPUT PHASE SIGNAL
2. GENERATE THE PLL INPUT FM SPECTRA
3. GENERATE THE PLL DEMODULATED SIGNAL
4. SELECT NEW FM MODULATOR / PLL PARAMETERS
5. SELECT NEW NTSC SIGNAL GENERATOR PARAMETERS
6. STORE THE CURRENT PARAMETERS (NTSCPARA.TXT)
7. RECALL THE STORED NTSC PARAMETERS (NTSCPARA.TXT)
8. DISPLAY THE STORED NTSC PARAMETERS (NTSCPARA.TXT)

ENTER SELECTION OR <RETURN> TO QUIT ----->

Menu selections #1, #2 and #3 each generate real data files representing the PLL input phase, the PLL input FM spectra and the PLL demodulated signal respectively. Selection #2 prompts the user for a noise floor (or default) for the log magnitude data. Selection #3 allows the user to specify FM
deemphasis to restore the original signal characteristics. Selections #4 and #5 return control to the FM MODULATOR/DEMODULATOR PARAMETER and the GET NTSC COEFFICIENTS menus respectively. Menu selections #6, #7 and #8 are included for storage, retrieval and displaying of the system parameters. A sample screen display for menu selection #8 is shown below. The parameters are displayed as they appear in the text file NTSCPARA.TXT.

**NTSC COLOR BAR GENERATOR**

<table>
<thead>
<tr>
<th>VIDEO FILTERS</th>
<th>PREEMPH.</th>
<th>PERCENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cutoff Freq. (MHz)</td>
<td># of Poles</td>
<td>(Y/N)</td>
</tr>
<tr>
<td>---------------</td>
<td>------------</td>
<td>---------</td>
</tr>
<tr>
<td>y(t)</td>
<td>q(t)</td>
<td>i(t)</td>
</tr>
</tbody>
</table>

**FM TRANSMITTER (VCO)**

<table>
<thead>
<tr>
<th>PEAK FREQ. DEVIATION (MHz)</th>
<th>MODULATION INDEX (MHz/volt)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.700</td>
<td>36.507</td>
</tr>
</tbody>
</table>

**FM DEMODULATOR (LINEAR PLL MODEL)**

<table>
<thead>
<tr>
<th>KdKv (1/sec)</th>
<th>T1 (nsec)</th>
<th>T2 (nsec)</th>
<th>Fn (MHz)</th>
<th>Damping Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2310E+09</td>
<td>26.005</td>
<td>10.670</td>
<td>15.000</td>
<td>0.70682</td>
</tr>
</tbody>
</table>

PRESS <RETURN> TO CONTINUE

The user may exit the program from the final menu by pressing <RETURN>. At any point in the program, the user may exit to the operating system by pressing CTRL-C.
Appendix 3. Hardware Schematics
NTSC Color Bar Generator

Block Diagram

RGB Signal Generator

Red

Green

Blue

NTSC Encoder

Video Output

Horz. Sync.

Horizontal Sync. Source
NTSC Encoder

[Diagram of NTSC Encoder circuit with components and connections labeled]
Timing Diagram

Note 1: Adjust R9 for $f_h = 15734.26\text{ Hz}$. Frequency may drift over time.

Note 2: Adjust R1 for approx. 9.5 $\mu\text{sec}$ delay as shown.

Note 3: Adjust R2 and R3 for 66.4 KHz, 50% duty cycle square wave as shown.

Note 4: Adjustment of R2, R3 and R1 will vary this period which should be approx. 1.5 $\mu\text{sec}$. (not critical)

Note 5: Adjust R7 for proper color burst position.

Note 6: Adjust R8 for 1 to 1.5 volts peak-to-peak video output.

Note 7: Adjust R4, R5 and R6 for 1 volt peak-to-peak for 100% saturated color bars.
NTSC Color Bar Pattern Generator Parts List

Resistors: (1/4 watt carbon composition)

R1,R4,R5,R6,R7,R8  10K ohm trimmer
R2  1K ohm trimmer
R3,R9  500 ohm trimmer
R10,R18,R24,R28,R29  10K ohm
R11  15K ohm
R12,R30  100 ohm
R13,R23  2.2K ohm
R14  6.8K ohm
R15  560 ohm
R16,R26,R27  1K ohm
R17,R19,R22  470 ohm
R20,R21  180 ohm
R25  47K ohm

Capacitors:

C1,C2,C34,C36  4.7 µF tantalum
C3,C9  10 pF ceramic
C4,C21,C22  .001 µF ceramic
C5  4700 pF mica
C6,C17,C27,C32  .01 µF ceramic
C7,C8,C11,C12,C13,C14,C28  10 µF electrolytic
C10,C15,C23,C24,C25,C26,C29,
C35,C37  .1 µF ceramic
C16  500 µF electrolytic
C18,C19  270 pF ceramic
C20  5-30 pF trimmer
C30  200 μF electrolytic
C31  .01 μF mylar
C33  1000 μF electrolytic

Inductors:
L1    RF choke (50 turns of # 32 wire on a T25-2 core)

Transistors:
D1,D2,D3,D4,D5,D6  1N914 silicon diode
Q1    2N2222 NPN general purpose

Integrated Circuits:
IC1    74LS123 dual monostable multivibrator
IC2,IC6  NE555 timer
IC3    74121 monostable multivibrator
IC4    7474 dual D flip-flop
IC5    MC1377 NTSC encoder (Motorola)
IC7    7805 +5 volt regulator
IC8    7812 +12 volt regulator

Miscellaneous:
XTAL1  3579545 Hz color burst crystal
CCIR Recommendation 405-1

Preemphasis Filter

Deemphasis Filter

R  75Ω
R1 275.8Ω
R3 20.4Ω
Cp 308.5 pF
Lp 17.35 μH

R  75Ω
R1 275.8Ω
R3 20.4Ω
Cd 891.7 pF
Ld 50.16 μH
FM Modulator

-5VDC Regulated

IC 600
MC1658

12
5

11
14

R604 330
R605 330

C607 .01

J601

70 MHz FM Output

Modulator Input

Input Bias

R601 820
R602 75
R603 75
C601 .01
C602 20

C603 .1
C604 10
C605 .01
C606 10
C609 .056
C608 5-30
L600 100

J600

116
PM Modulator Parts List

Resistors: (1/4 watt carbon composition)

R600 2K ohm multiturn trimmer
R601 820 ohm
R602,R603 75 ohm
R604,R605 330 ohm

Capacitors:

C600 4.7 µF tantalum
C601,C605,C607 .01 µF ceramic
C602 20 µF tantalum
C603 .1 µF ceramic
C604,C606 10 µF 50V electrolytic
C608 5-30 pF trimmer
C609 .056 µF ceramic chip

Inductors:

L600 100 µH RF choke

Integrated Circuits:

IC600 MC1658 voltage controlled multivibrator (Motorola)

Connectors:

J600,J601 BNC female
PLL Video Demodulator

*Jumper 1 shown in normal position. Connect 1 to 2 for free run, open loop testing.
**Jumper 2 shown in normal position. Connect 2 to 3 for phase error measurements.
PLL Video Demodulator Parts List

Resistors: (1/4 watt carbon composition)

R100 5K ohm trimmer
R101 10K ohm
R102,R109 820 ohm
R103 100 ohm trimmer
R104 2.7K ohm
R105,R115 100 ohm
R106 140 ohm
R107,R108 680 ohm
R110,R111 330 ohm
R112,R113,R114,R116,R118, R122,R125,R126,R127 75 ohm
R117 560 ohm
R119,R120 15K ohm
R121,R124 470 ohm
R123 51 ohm

Capacitors:

C100,C121 .1 µF ceramic
C101,C106,C119,C130,C132, C134,C136,C138 .01 µF ceramic
C102,C109,C118,C129,C131, C133 10 µF tantalum
C103,C104,C110,C111,C112, C114,C115,C116,C117,C135 .056 µF ceramic chip
C105,C108 5-30 pF trimmer
C107 33 pF ceramic
C113 470 pF ceramic
<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C120</td>
<td>270 pF ceramic</td>
</tr>
<tr>
<td>C122, C123, C124</td>
<td>2000 pF ceramic</td>
</tr>
<tr>
<td>C125</td>
<td>100 µF 50V electrolytic</td>
</tr>
<tr>
<td>C126</td>
<td>4.7 µF tantalum</td>
</tr>
<tr>
<td>C127</td>
<td>6.8 µF tantalum</td>
</tr>
<tr>
<td>C128</td>
<td>1000 pF ceramic</td>
</tr>
<tr>
<td>C137</td>
<td>22 pF silver mica</td>
</tr>
</tbody>
</table>

**Inductors:**
- L100, L101: 1/8" D, 1/4" L ferrite bead
- L102: .96 µH (16 turns of #32 wire on a T25-2 core)
- L103: 100 µH RF choke

**Transistors:**
- D100, D101, D102, D103: 1N914 silicon diode
- Q100, Q101, Q102: 2N918 NPN high frequency
- Q103, Q104: 2N2222 NPN general purpose

**Integrated Circuits:**
- IC100: MC1658 voltage controlled multivibrator (Motorola)
- IC101: MC12002 analog mixer (Motorola)
- IC102: LM336 precision +5 volt reference
- IC103: LM7812 +12 volt regulator
- IC104: LM7805 +5 volt regulator

**Connectors:**
- J100: F coaxial female
J101, J106  Single conductor push-on
J102, J103, J104, J105  BNC female
THE RMS PHASE ERROR OF A PHASE-LOCKED LOOP FM DEMODULATOR FOR STANDARD NTSC VIDEO

by

DALE F. DUBBERT
B.S., Kansas State University, 1984

AN ABSTRACT OF A MASTER'S REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Electrical and Computer Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1986
Abstract

A second order phase-locked loop (PLL) is a well known scheme for the demodulation of FM signals. A major consideration in the design of a PLL FM demodulator is to keep the phase error relatively small thus assuring minimum demodulator output distortion due to phase detector nonlinearities and, in extreme cases, loss of phase lock. The loop bandwidth must be selected to both maintain a small phase error (wide loop bandwidth) and to keep the demodulator output signal to noise ratio as large as possible (narrow loop bandwidth). In the absence of a qualitative signal-to-noise ratio analysis, the procedure is to select the second order loop natural frequency to be as low as possible. Below this frequency, the loop phase error increases drastically.

This paper documents a study of the RMS phase error performance of a second order phase-locked loop employed as a satellite FM video demodulator. The phase error for the PLL was both computer calculated and experimentally measured using a standard NTSC color bar pattern as a test signal. The experimental and calculated results illustrate the relationship between RMS phase error and the second order PLL natural frequency. Both sets of results were found to be well correlated, thus providing important information needed for an optimum demodulator design.