Final Report

Project A-1379

Study of Mathematical Modeling of Communication Systems, Transponders and Receivers

J. R. Walsh, R. D. Wetherington and L. D. Holland

Contract NAS8-28148

19 November 1972

Prepared for
National Aeronautics & Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama

(Georgia Inst. of Tech.) 201 p HC $12.25

Engineering Experiment Station
GEORGIA INSTITUTE OF TECHNOLOGY
Atlanta, Georgia
FINAL REPORT

Project A-1379

STUDY OF MATHEMATICAL MODELING OF COMMUNICATION SYSTEMS TRANSPONDERS AND RECEIVERS

J. R. Walsh, R. D. Wetherington and L. D. Holland

CONTRACT NAS8-28148

19 November 1972

Prepared for

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
GEORGE C. MARSHALL SPACE FLIGHT CENTER
MARSHALL SPACE FLIGHT CENTER, ALABAMA
ABSTRACT

This report presents the results of work on the modeling of communication receivers at both the circuit detail level and at the block level. The largest effort was devoted to developing new models at the block modeling level. The available effort did not permit full development of all of the block modeling concepts envisioned, but idealized blocks were developed for signal sources, a variety of filters, limiters, amplifiers, mixers, and demodulators. These blocks were organized into an operational computer simulation of communications receiver circuits identified as the Frequency And Time Circuit Analysis Technique (FATCAT). The simulation operates in both the time and frequency domains, and permits output plots or listings of either frequency spectra or time waveforms from any model block. Transfer between domains is handled with a fast Fourier transform algorithm.

A separate block model effort was devoted to developing block models for use with the circuit simulation MIMIC.

Two efforts in modeling at the circuit detail level were also carried out. One of these demonstrated the feasibility of interfacing the time-domain analysis program CIRCUS with a frequency-domain analysis program to provide a more powerful model. The other effort demonstrated the use of ECAP to determine input parameters for CIRCUS which are otherwise difficult to determine.

Computer listings of the software developed and examples of use of the programs are included.
FOREWORD

This report was prepared at the Engineering Experiment Station at the Georgia Institute of Technology for the Astronics Laboratory of Marshall Space Flight Center under Contract NAS8-28148. The work was carried out under the direct supervision of Mr. J. R. Walsh, Project Director, and under the general supervision of Mr. D. W. Robertson, Chief of the Communications Division. The report describes the results of a one-year effort on the modeling of communications systems.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>I. INTRODUCTION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>II. MATHEMATICAL MODELING OF COMMUNICATIONS CIRCUITS ON A CIRCUIT DETAIL BASIS</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Interfacing CIRCUS and the FFT</td>
<td>5</td>
</tr>
<tr>
<td>1. Time Domain vs Frequency Domain Analysis</td>
<td>5</td>
</tr>
<tr>
<td>2. Description of Interface and Software</td>
<td>6</td>
</tr>
<tr>
<td>3. Example of Using the CIRCUS-Frequency Domain Interface Program</td>
<td>8</td>
</tr>
<tr>
<td>B. Use of ECAP for Transfer Function Evaluation</td>
<td>19</td>
</tr>
<tr>
<td>C. Conclusions and Recommendations</td>
<td>25</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>III. MATHEMATICAL MODELING OF COMMUNICATION SYSTEMS ON A BLOCK BASIS</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. General Description</td>
<td>29</td>
</tr>
<tr>
<td>B. Program Control</td>
<td>30</td>
</tr>
<tr>
<td>1. Block Input Commands</td>
<td>30</td>
</tr>
<tr>
<td>a. Sources</td>
<td>31</td>
</tr>
<tr>
<td>b. Filters</td>
<td>32</td>
</tr>
<tr>
<td>c. Demodulators</td>
<td>33</td>
</tr>
<tr>
<td>d. Other Blocks</td>
<td>33</td>
</tr>
<tr>
<td>2. Control Commands</td>
<td>34</td>
</tr>
<tr>
<td>3. Special Commands</td>
<td>37</td>
</tr>
<tr>
<td>C. Simulation Examples using FATCAT</td>
<td>39</td>
</tr>
<tr>
<td>1. AM Receiver</td>
<td>39</td>
</tr>
<tr>
<td>2. FM Receiver</td>
<td>52</td>
</tr>
<tr>
<td>D. Conclusions</td>
<td>57</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IV. TIME DOMAIN SIMULATION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Introduction</td>
<td>71</td>
</tr>
<tr>
<td>B. MIMIC: A Continuous System Simulation Language</td>
<td>72</td>
</tr>
<tr>
<td>C. TIMSIM: A User-Oriented Communications System Block Diagram Simulation Program</td>
<td>79</td>
</tr>
<tr>
<td>1. Subprogram Derivation</td>
<td>79</td>
</tr>
<tr>
<td>2. TIMSIM Subprogram Library</td>
<td>81</td>
</tr>
<tr>
<td>3. Applications</td>
<td>81</td>
</tr>
<tr>
<td>D. Conclusions</td>
<td>95</td>
</tr>
<tr>
<td>TABLE OF CONTENTS (Continued)</td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td></td>
</tr>
<tr>
<td>V. REFERENCES ..................</td>
<td>96</td>
</tr>
<tr>
<td>APPENDICES:</td>
<td></td>
</tr>
<tr>
<td>APPENDIX A: ....................</td>
<td>97</td>
</tr>
<tr>
<td>APPENDIX B: ....................</td>
<td>101</td>
</tr>
<tr>
<td>APPENDIX C: ....................</td>
<td>117</td>
</tr>
<tr>
<td>APPENDIX D: ....................</td>
<td>179</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Test Circuit for the CIRCUS Frequency Domain Interface Program</td>
</tr>
<tr>
<td>2</td>
<td>Execution of CIRCUS for Test Circuit</td>
</tr>
<tr>
<td>3</td>
<td>Execution of Interface Program</td>
</tr>
<tr>
<td>4</td>
<td>Spectrum of Data Set 3</td>
</tr>
<tr>
<td>5</td>
<td>Time Waveform of Data Set 3</td>
</tr>
<tr>
<td>6</td>
<td>Request for Calcomp Plot</td>
</tr>
<tr>
<td>7</td>
<td>Calcomp Plot of the Frequency Spectrum of the Time Waveform Shown in Figure 5</td>
</tr>
<tr>
<td>8</td>
<td>Circuit Diagram of CCS Telemetry Filter</td>
</tr>
<tr>
<td>9</td>
<td>Circuit Description in ECAP Format</td>
</tr>
<tr>
<td>10</td>
<td>Comparison of Response Calculation for CCS Telemetry Filter</td>
</tr>
<tr>
<td>11</td>
<td>Sample of ECAP Output</td>
</tr>
<tr>
<td>12</td>
<td>Block Diagram of Amplitude Modulation Receiver and FATCAT Model of Receiver</td>
</tr>
<tr>
<td>13</td>
<td>Start of FATCAT Run for AM Receiver, Showing Spectrum at the Output of the Signal Generator</td>
</tr>
<tr>
<td>14</td>
<td>Listing of the Frequency Function at the Output of the Signal Generator</td>
</tr>
<tr>
<td>15</td>
<td>Listing of Frequency Function at the Output of the First Bandpass Filter</td>
</tr>
<tr>
<td>16</td>
<td>Spectrum at the Output of the First Bandpass Filter</td>
</tr>
<tr>
<td>17</td>
<td>Spectrum at the Output of the Ideal Multiplier</td>
</tr>
<tr>
<td>18</td>
<td>Spectrum at the Output of the Second Bandpass Filter</td>
</tr>
<tr>
<td>19</td>
<td>Spectrum at the Output of the Amplifier</td>
</tr>
<tr>
<td>20</td>
<td>Spectrum at the Output of the AM Demodulator</td>
</tr>
<tr>
<td>21</td>
<td>Time Waveform at the Output of the AM Demodulator</td>
</tr>
<tr>
<td>22</td>
<td>Listing of Time Function at the Output of the AM Demodulator</td>
</tr>
<tr>
<td>23</td>
<td>Spectrum at the Output of the AM Demodulator</td>
</tr>
<tr>
<td>24</td>
<td>Spectrum at the Output of the Low Pass Filter</td>
</tr>
<tr>
<td>25</td>
<td>Block Diagram for FATCAT Analysis of an FM Receiver</td>
</tr>
</tbody>
</table>
LIST OF FIGURES (Continued)

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>Start of FATCAT Run for FM Receiver, Showing Spectrum at the Output of the Signal Generator</td>
<td>58</td>
</tr>
<tr>
<td>27</td>
<td>Spectrum at the Output of the First Bandpass Filter</td>
<td>59</td>
</tr>
<tr>
<td>28</td>
<td>Spectrum at the Output of the Ideal Multiplier</td>
<td>60</td>
</tr>
<tr>
<td>29</td>
<td>Spectrum at the Output of the Second Bandpass Filter</td>
<td>61</td>
</tr>
<tr>
<td>30</td>
<td>Spectrum at the Output of the Amplifier</td>
<td>62</td>
</tr>
<tr>
<td>31</td>
<td>Spectrum at the Output of the FM Demodulator</td>
<td>63</td>
</tr>
<tr>
<td>32</td>
<td>Time Waveform at the Output of the FM Demodulator</td>
<td>64</td>
</tr>
<tr>
<td>33</td>
<td>Listing of the Frequency Function at the Output of the FM Demodulator</td>
<td>65</td>
</tr>
<tr>
<td>34</td>
<td>Spectrum at the Output of the Low Pass Filter</td>
<td>66</td>
</tr>
<tr>
<td>35</td>
<td>Time Waveform at the Output of the Low Pass Filter</td>
<td>67</td>
</tr>
<tr>
<td>36</td>
<td>Listing of the Frequency Function at the Output of the Low Pass Filter</td>
<td>68</td>
</tr>
<tr>
<td>37</td>
<td>Detailed MIMIC Block Diagram for a System Described by $\dot{x} + x + x = 0$</td>
<td>74</td>
</tr>
<tr>
<td>38</td>
<td>MIMIC Instructions for Simulation of the System of Figure 37</td>
<td>74</td>
</tr>
<tr>
<td>39</td>
<td>Block Diagram of Feedback Example</td>
<td>76</td>
</tr>
<tr>
<td>40</td>
<td>MIMIC Simulation of System of Figure 39</td>
<td>76</td>
</tr>
<tr>
<td>41</td>
<td>Subprogram Example</td>
<td>78</td>
</tr>
<tr>
<td>42</td>
<td>Single-Tuned Filter Flow Diagram</td>
<td>80</td>
</tr>
<tr>
<td>43</td>
<td>Hypothetical AM Communication System</td>
<td>84</td>
</tr>
<tr>
<td>44</td>
<td>TIMSIM Listing, Hypothetical Communication System</td>
<td>85</td>
</tr>
<tr>
<td>45</td>
<td>TIMSIM Results, Hypothetical AM Communication System</td>
<td>88</td>
</tr>
<tr>
<td>46</td>
<td>Plot of TIMSIM Results for AM System</td>
<td>90</td>
</tr>
<tr>
<td>47</td>
<td>Hypothetical Automatic Gain Control System</td>
<td>91</td>
</tr>
<tr>
<td>48</td>
<td>TIMSIM Listing, Automatic Gain Control System</td>
<td>92</td>
</tr>
<tr>
<td>49</td>
<td>TIMSIM Results, AGC System</td>
<td>94</td>
</tr>
</tbody>
</table>
I. INTRODUCTION

This report discusses the results of a one-year effort directed toward the modeling of communications receivers. This effort followed a program carried out during the previous year under contract NAS8-20054 in which transmitter modeling was considered. Most of the modeling work carried out on the former program was directed toward modeling at the circuit detail level, and the simulation program CIRCUS was used as the primary analysis tool. Accomplishments under that program have been documented in previous reports [1,2].

On the current program, modeling of communications receiver circuits has been carried out on both the circuit detail level and the block modeling level, with emphasis on the latter. Several existing circuit simulation programs have been investigated and three of these (CIRCUS, ECAP, and MIMIC) were used in the modeling efforts reported herein. In addition, using a block modeling approach, work on a new circuit simulation program was undertaken and the program was developed to an operational state that provides a highly useful circuit analysis tool. However, the present form of the program is far from realizing the potential that the approach offers.

Two lines of effort were pursued in circuit detail modeling. One of these was an investigation of the feasibility of interfacing CIRCUS with some other circuit analysis program which operates in the frequency domain, thereby creating a more powerful analysis tool. The other effort was concerned with using ECAP as a basic tool for providing certain input parameters for CIRCUS which are difficult to determine otherwise. The results of both investigations were promising.

The need for interfacing CIRCUS with some other program arises from the fact that CIRCUS operates entirely in the time domain. Processing of signals requires step-by-step integration of the time function. Such a procedure is necessary when dealing with nonlinear portions of a circuit; when linear circuits with known transfer functions are involved, the effect of the circuit is much more readily computed in the frequency domain. Thus a model with the capability of processing nonlinear circuits
in the time domain, and linear portions in the frequency domain would be very attractive. Since CIRCUS is a very powerful time domain simulation, it would provide a good starting point for creating a two-domain model if it could be successfully interfaced with a frequency domain model.

The investigations conducted were really in the nature of feasibility investigations. Methods were developed for extracting the computed time functions from CIRCUS, converting them to the frequency domain, and displaying either time or frequency functions. This work is discussed in Section II and all software developed in this effort is listed in Appendices A and B.

The work with ECAP was directed toward using the a-c analysis capabilities of ECAP to investigate circuit parameter effects on the transfer functions of certain circuits, and thereby determine suitable parameter values for entry into CIRCUS. There are certain parameters required in specifying circuits to CIRCUS which are not easily estimated. One example is the values of coupling coefficients in coupled circuits. Unless reasonably good values for these parameters can be determined prior to entry into CIRCUS, interpreting the program's output may be very nearly impossible if it runs; sometimes the program will not even operate if the values are unrealistic.

The investigations reported herein show that ECAP is a useful tool for determining parameter values in many cases. Details of the investigation are also discussed in Section II.

The major effort in this program was devoted to circuit models based on a block modeling (as opposed to circuit detail modeling) approach. The block modeling work can also be divided into two lines of effort. The primary effort was devoted to developing models for circuit blocks and combining them to create an entirely new simulation program. The other effort consisted of investigating the use of the simulation program MIMIC in modeling circuits with feedback loops on a block modeling basis.

Development of the new simulation program was directed toward modeling each subsystem (mixers, filters, amplifiers, etc.) as a single block in order to obtain an efficient program for rapid analysis of communications receivers. The simulation was planned from the beginning to operate
in both the time and frequency domains. A representation of the signal is stored in a complex data array, and all processing blocks operate on this array. The array contents at any given time may be either a discrete spectrum representing the frequency function, or data points representing samples of a time waveform. The array contents are transformed from one domain to another as needed by a fast Fourier transform (FFT) algorithm.

The block model for each subsystem can be modeled in either the time domain or the frequency domain. Blocks have been created for signal sources, a variety of filters, mixers, limiters, amplifiers, and demodulators. Other needed software was developed to create an operational simulation identified as the Frequency and Time Circuit Analysis Technique (FATCAT). Control software was designed for conversational operation from a real time computer terminal; it is readily adaptable to batch processing, however.

The program was developed and completely implemented on a Univac-1108 computer at Georgia Tech. Required modifications were then made to adapt the program to a SIGMA-5 computer at Marshall Space Flight Center. The program is discussed in Section III and descriptions and listings of all software for the Univac-1108 version are given in Appendix C. Listings of the SIGMA-5 versions of those routines requiring modification are given in Appendix D.

Although the version of FATCAT presented here was brought to an operational state and has proved to be a highly useful simulation, it is by no means the ultimate simulation of its type. Many ideas exist for improving the program which could not be investigated during the time available. Ideas for improvement include changes to upgrade some of the existing model blocks, creation of new model blocks, and additional features that could be added to the control software which would make the simulation even easier to use. Some specific examples of improvements that should be developed are pointed out in Section IIID.

The other block modeling effort carried out under this program made use of the simulation program MIMIC, a dynamic system simulation program. MIMIC accepts user inserted "models" in the form of equations, and this feature was investigated as a method of modeling circuit blocks. The investigation is discussed in Section IV.
II. MATHEMATICAL MODELING OF COMMUNICATIONS CIRCUITS
ON A CIRCUIT DETAIL BASIS

A. Interfacing CIRCUS and the FFT

1. Time Domain vs Frequency Domain Analysis

One of the major efforts related to circuit detail modeling was devoted to interfacing CIRCUS with other programs so that frequency domain representations of signals could be obtained. CIRCUS itself operates entirely in the time domain. Processing the signal requires step-by-step integration, often with very small steps. Although the process can be time consuming, it is a good method for performing analyses that must be performed in the time domain, such as analyzing nonlinear circuits or calculating the transient response of any circuit.

If only the steady state solution is desired for linear circuits, frequency domain analysis can be applied. Since frequency domain analysis is generally much less time consuming, and also since the steady state response is of prime interest in most circuit investigations, it would be preferable to analyze linear circuits in the frequency domain. However, entire equipments (such as a communications receiver) cannot usually be analyzed entirely in the frequency domain since they usually contain nonlinearities. The ideal solution would be to have an analysis program that operates in both domains and to have the ability to transfer from one domain to another.

The investigation reported here was directed toward investigating the possibility of combining circuit detail models which operate in the two domains, and thus create a single more powerful model with more efficient operation. The job of actually creating such a model was beyond the scope of this investigation, and only a feasibility study has been conducted.

Specifically, the task undertaken was to develop an interface with CIRCUS which would permit extraction of computed time waveforms. These signals could then be operated on with a fast Fourier transform (FFT) to obtain frequency domain representations of the signals. Such an interface has been constructed, along with some supporting software that
permits recovery of the CIRCUS-generated signals, transforming between domains, and output of results. Details of the interface and software are described below, followed by sample runs.

2. Description of Interface and Software

For purposes of this feasibility study, the modifications made internally in CIRCUS were kept to a minimum. The immediate objective was to gain access to the computed time waveform data points, and to write these into a file external to CIRCUS so that the data would be stored when the run terminated. Other software was then constructed to access the stored data file and process it.

After examining several of the CIRCUS subroutines, it appeared that the desired data could be obtained easily by constructing a subroutine to be called by the PLOT statement. As a working procedure, subroutine PLOTER was disabled (by commenting out its call in subroutine LINK6A), and a new subroutine, PLTDTA, was constructed to extract the data. PLTDTA writes the data out on one of the FORTRAN output units (currently set at 19) and then terminates. This form of modification permitted testing without the necessity of revising the CIRCUS command structure. The PLOT statement could be used to specify the outputs wanted, and outputs from as many nodes as desired could be obtained. Thus the changes in CIRCUS were limited to construction of PLTDTA plus minor modifications to LINK6A. The modifications made in LINK6A were (1) deleting the calls to PLOTER by inserting comments on line, and (2) inserting calls to PLTDTA at the appropriate point. Listings of PLTDTA and the modified version of LINK6A are given in Appendix A.

The output generated by PLTDTA consists of two parameters and three arrays. In order these are: (1) the parameter NPNT specifying the number of points per data set; (2) the parameter NPLOTS specifying the number of data sets to be outputed; (3) the array TIME containing NPNT entries of the times corresponding to output data points; (4) the array PLOT containing NPLOTS sets of output data, each consisting of NPNT values of a node voltage, an element current, or whatever variable was
called for in the PLOT statement;* and (5) the array TITLE containing
the title of the run (not used by the current version of the interface
program).

In using the program, the modified version of CIRCUS is run with a
PLOT statement included that has a list of the variables to be written
out. When the computations are finished the CIRCUS run is terminated.
The interface program can then be activated to call the data file and
operate on it; the call is non-destructive and the same data file can
be processed many times by the interface program.

There is one specific requirement that the user must be aware of
and provide for. The FFT is designed to work only with data sets of N
points where N is a power of two (N = 2^{IGAM}, where IGAM is an integer).
Since the theory underlying the FFT requires that it be applied only
to periodic signals, the user must pre-determine the period of the out-
put waveform to be generated by CIRCUS. The period must then be divided
by the chosen value of N (with N = 2^{IGAM}) to determine the time step
at which outputs will be generated. This procedure insures that exactly
N consecutive outputs will represent exactly one period of the output
waveform.

While the time step size must be carefully determined, the overall
time interval for which outputs are generated is not critical. More
than N points can be outputed so long as exactly N consecutive points
represent one period. In general it will be desirable to output more
than N points so that if the first few contain transient effects they
can be discarded.

The interface program, PLT, is used to access the data file and dis-
play the data in either the time domain or the frequency domain. Display
capabilities include printed listings and plotted time waveforms or fre-
quency spectra. Two types of plot routines are available, printer type
plots (scaled not to exceed 72 columns so that they can be displayed by
a teletype terminal), and Calcomp plots (frequency spectra only).

*This list can contain any of the output variables specified in the CIRCUS
users manual [3].
In the next section a specific example of using the program is shown, and the overall operation is further explained. A listing of all software used in the program is given in Appendix B.

3. Example of Using the CIRCUS-Frequency Domain Interface Program

Consider the simple transistor feedback amplifier shown in Figure 1. This amplifier, when operating in its linear region, would have a gain approximately equal to the ratio of the collector to emitter resistances. The drive has been set to a level such that the amplifier operates in a nonlinear region, thereby producing distortion in the output signal. A sample execution of the CIRCUS-frequency domain interface program for this circuit is presented to demonstrate the capabilities of the program.

Before executing CIRCUS, the file to accept the time waveform data from CIRCUS must be declared and linked to logical unit 19. The setup on the Univac-1108 is as follows:

```
@ASG,A PFILE.
@USE 19, PFILE
```

PFILE could be either a tape file or a mass storage file; as used here it is a Fastrand (mass storage) file. Execution of CIRCUS with a plot statement containing the data to be plotted will write these data on PFILE. An execution of CIRCUS for the simple circuit shown in Figure 1 is shown in Figure 2.

To use the transform and plotting program PLT, the file containing the CIRCUS data must be assigned to the run by the statements:

```
@ASG,A PFILE.
@USE 19, PFILE
```

The command to execute the plotting program produces several messages relative to the number and size of the data sets generated in CIRCUS and to the number of points in a transform. Figure 3 shows a sample execution of the transform and plotting program. The first messages received
Figure 1. Test Circuit for the CIRCUS Frequency Domain Interface Program.
START      51
#ADD CD1,DATA17
'SINGLE STAGE TEST CIRCUIT'
R1, 1, 2, 100.
R2, 3, 7, 8.2E3
R3, 0, 3, 2.2E3
R4, 4, 7, 1.E3
R5, 5, 6, 68.
R6, 6, 6, 270.
C1, 2, 3, 1.E-8
C2, 0, 6, 1.E-6
VSI, 0, 7, 12.
T1, 3, 4, 5, 2N697
SV1, 0, 1, 0, 0.5, 0., 2.E-6
DEVICE PARAMETERS
TRANSISTOR, 2N697, NPN,
RB, 70., HC, 1.5, RE, .001, A1, 54.E-12,
PHI1, .9, N1, .37, A2, 30.3E-12, PHI2, .9,
N2, .36, IES, 4.47E-14, ICS, 2.24E-13, THETAN, 38.8,
THETA1, 35.5
    BN, 0., 1.E-5, 1., 68.
    B1, 0., 1.E-5, 1., 4.7
TCN, .001, 9.93E-10
TC1, .001, 91E-7
END
INTERVALS, .125E-6, 8.E-6
PRINT, VN1, VN2, VN4, IBT1, ICT1
PLOT, VN1, VN2, VN4, IBT1, ICT1
HOLD FINAL CONDITIONS
EXECUTE
EXECUTE
LINK5 50

Figure 2. Execution of CIRCUS for Test Circuit.
Figure 2. (Continued).

### TIME (USEC)

<table>
<thead>
<tr>
<th>TIME (USEC)</th>
<th>VN1</th>
<th>VN2</th>
<th>VN4</th>
<th>IBT1</th>
<th>ICT1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>1.676-08</td>
<td>6.91</td>
<td>4.491-05</td>
<td>5.094-03</td>
</tr>
<tr>
<td>1.125</td>
<td>1.91</td>
<td>1.448</td>
<td>5.59</td>
<td>4.245-04</td>
<td>6.407-03</td>
</tr>
<tr>
<td>2.500</td>
<td>3.54</td>
<td>2.96</td>
<td>3.74</td>
<td>4.858-04</td>
<td>8.264-03</td>
</tr>
<tr>
<td>5.000</td>
<td>5.00</td>
<td>4.30</td>
<td>2.19</td>
<td>5.375-04</td>
<td>9.806-03</td>
</tr>
<tr>
<td>7.500</td>
<td>3.54</td>
<td>3.54</td>
<td>2.15</td>
<td>-1.160-04</td>
<td>9.853-03</td>
</tr>
<tr>
<td>8.750</td>
<td>1.91</td>
<td>2.41</td>
<td>2.84</td>
<td>-5.401-04</td>
<td>9.156-03</td>
</tr>
</tbody>
</table>

### TIME (USEC)

<table>
<thead>
<tr>
<th>TIME (USEC)</th>
<th>VN1</th>
<th>VN2</th>
<th>VN4</th>
<th>IBT1</th>
<th>ICT1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.000</td>
<td>1.589-08</td>
<td>4.633-02</td>
<td>5.16</td>
<td>-3.996-04</td>
<td>6.241-03</td>
</tr>
<tr>
<td>1.125</td>
<td>2.91</td>
<td>1.44</td>
<td>8.46</td>
<td>-2.984-04</td>
<td>3.536-03</td>
</tr>
<tr>
<td>1.250</td>
<td>3.54</td>
<td>2.96</td>
<td>10.6</td>
<td>-2.144-04</td>
<td>1.443-03</td>
</tr>
<tr>
<td>1.375</td>
<td>4.62</td>
<td>4.00</td>
<td>11.7</td>
<td>-1.022-04</td>
<td>2.884-04</td>
</tr>
<tr>
<td>1.500</td>
<td>5.00</td>
<td>4.31</td>
<td>11.9</td>
<td>3.461-06</td>
<td>5.021-05</td>
</tr>
<tr>
<td>1.625</td>
<td>4.62</td>
<td>4.36</td>
<td>11.9</td>
<td>6.266-05</td>
<td>9.485-05</td>
</tr>
<tr>
<td>1.750</td>
<td>3.54</td>
<td>3.43</td>
<td>11.4</td>
<td>1.624-04</td>
<td>6.221-04</td>
</tr>
<tr>
<td>1.875</td>
<td>-1.91</td>
<td>9.95</td>
<td>2.823-04</td>
<td>2.054-03</td>
<td></td>
</tr>
<tr>
<td>2.000</td>
<td>-3.179-08</td>
<td>-2.918-02</td>
<td>7.91</td>
<td>3.838-04</td>
<td>4.088-03</td>
</tr>
<tr>
<td>2.125</td>
<td>4.91</td>
<td>1.45</td>
<td>5.72</td>
<td>4.551-04</td>
<td>6.282-03</td>
</tr>
<tr>
<td>2.250</td>
<td>3.54</td>
<td>2.96</td>
<td>3.77</td>
<td>4.886-04</td>
<td>8.228-03</td>
</tr>
<tr>
<td>2.375</td>
<td>4.62</td>
<td>4.00</td>
<td>2.40</td>
<td>4.794-04</td>
<td>9.598-03</td>
</tr>
<tr>
<td>2.500</td>
<td>5.00</td>
<td>4.31</td>
<td>2.19</td>
<td>5.303-04</td>
<td>9.806-03</td>
</tr>
<tr>
<td>2.625</td>
<td>-1.91</td>
<td>2.17</td>
<td>3.332-04</td>
<td>9.832-03</td>
<td></td>
</tr>
<tr>
<td>2.750</td>
<td>3.54</td>
<td>3.95</td>
<td>2.13</td>
<td>-1.222-04</td>
<td>9.852-03</td>
</tr>
<tr>
<td>2.875</td>
<td>1.91</td>
<td>2.40</td>
<td>2.90</td>
<td>-5.305-04</td>
<td>9.102-03</td>
</tr>
<tr>
<td>3.000</td>
<td>2.865-07</td>
<td>4.604-02</td>
<td>5.79</td>
<td>-3.955-04</td>
<td>6.206-03</td>
</tr>
<tr>
<td>3.125</td>
<td>-1.91</td>
<td>-1.44</td>
<td>8.49</td>
<td>-2.976-04</td>
<td>3.514-03</td>
</tr>
<tr>
<td>3.250</td>
<td>-3.54</td>
<td>-3.06</td>
<td>10.6</td>
<td>-2.141-04</td>
<td>1.424-03</td>
</tr>
<tr>
<td>3.375</td>
<td>-4.62</td>
<td>-4.19</td>
<td>11.7</td>
<td>-1.013-04</td>
<td>2.773-04</td>
</tr>
<tr>
<td>3.500</td>
<td>-5.00</td>
<td>-4.66</td>
<td>12.0</td>
<td>2.678-06</td>
<td>4.764-05</td>
</tr>
<tr>
<td>3.625</td>
<td>-4.62</td>
<td>-4.35</td>
<td>11.9</td>
<td>6.238-05</td>
<td>8.995-05</td>
</tr>
<tr>
<td>3.750</td>
<td>-3.54</td>
<td>-3.42</td>
<td>11.4</td>
<td>1.616-04</td>
<td>6.911-04</td>
</tr>
<tr>
<td>3.875</td>
<td>-1.91</td>
<td>-2.00</td>
<td>9.96</td>
<td>2.817-04</td>
<td>2.036-03</td>
</tr>
<tr>
<td>4.000</td>
<td>-4.212-07</td>
<td>-2.903-02</td>
<td>7.93</td>
<td>3.832-04</td>
<td>4.070-03</td>
</tr>
<tr>
<td>4.125</td>
<td>1.91</td>
<td>1.45</td>
<td>5.74</td>
<td>4.544-04</td>
<td>6.263-03</td>
</tr>
<tr>
<td>4.250</td>
<td>3.54</td>
<td>2.96</td>
<td>3.79</td>
<td>4.876-04</td>
<td>8.211-03</td>
</tr>
<tr>
<td>4.375</td>
<td>4.62</td>
<td>4.00</td>
<td>2.42</td>
<td>4.774-04</td>
<td>9.584-03</td>
</tr>
<tr>
<td>4.500</td>
<td>5.00</td>
<td>4.32</td>
<td>2.19</td>
<td>5.247-04</td>
<td>9.805-03</td>
</tr>
<tr>
<td>4.625</td>
<td>4.62</td>
<td>4.15</td>
<td>2.17</td>
<td>3.278-04</td>
<td>9.832-03</td>
</tr>
<tr>
<td>4.750</td>
<td>3.54</td>
<td>3.56</td>
<td>2.15</td>
<td>-1.271-04</td>
<td>9.852-03</td>
</tr>
<tr>
<td>4.875</td>
<td>1.91</td>
<td>2.39</td>
<td>2.94</td>
<td>-5.232-04</td>
<td>9.061-03</td>
</tr>
<tr>
<td>5.000</td>
<td>5.563-07</td>
<td>4.583-02</td>
<td>5.82</td>
<td>-3.925-04</td>
<td>6.176-03</td>
</tr>
<tr>
<td>5.125</td>
<td>-1.91</td>
<td>-1.44</td>
<td>8.50</td>
<td>-2.970-04</td>
<td>3.495-03</td>
</tr>
<tr>
<td>5.250</td>
<td>-3.54</td>
<td>-3.06</td>
<td>10.6</td>
<td>-2.139-04</td>
<td>1.409-03</td>
</tr>
<tr>
<td>5.375</td>
<td>-4.62</td>
<td>-4.19</td>
<td>11.7</td>
<td>-1.007-04</td>
<td>2.683-04</td>
</tr>
<tr>
<td>5.500</td>
<td>-5.00</td>
<td>-4.65</td>
<td>12.0</td>
<td>2.095-06</td>
<td>4.604-05</td>
</tr>
<tr>
<td>5.625</td>
<td>-4.62</td>
<td>-4.35</td>
<td>11.9</td>
<td>6.217-05</td>
<td>8.619-05</td>
</tr>
<tr>
<td>5.750</td>
<td>-3.54</td>
<td>-3.42</td>
<td>11.4</td>
<td>1.609-04</td>
<td>5.989-04</td>
</tr>
<tr>
<td>5.875</td>
<td>-1.91</td>
<td>-2.00</td>
<td>9.98</td>
<td>2.812-04</td>
<td>2.022-03</td>
</tr>
<tr>
<td>6.125</td>
<td>1.91</td>
<td>1.45</td>
<td>5.75</td>
<td>4.538-04</td>
<td>6.249-03</td>
</tr>
</tbody>
</table>

---

**SINGLE STAGE TEST CIRCUIT**
### Single Stage Test Circuit

<table>
<thead>
<tr>
<th>TIME (USEC)</th>
<th>V1N1</th>
<th>V1N2</th>
<th>V1N4</th>
<th>IB1T</th>
<th>ICT1</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.250</td>
<td>354</td>
<td>296</td>
<td>3.80</td>
<td>4.867E-04</td>
<td>8.197E-03</td>
</tr>
<tr>
<td>6.375</td>
<td>462</td>
<td>400</td>
<td>2.43</td>
<td>4.759E-04</td>
<td>9.573E-03</td>
</tr>
<tr>
<td>6.500</td>
<td>500</td>
<td>432</td>
<td>2.19</td>
<td>5.202E-04</td>
<td>9.805E-03</td>
</tr>
<tr>
<td>6.625</td>
<td>462</td>
<td>415</td>
<td>2.17</td>
<td>3.235E-04</td>
<td>9.832E-03</td>
</tr>
<tr>
<td>6.750</td>
<td>354</td>
<td>356</td>
<td>2.15</td>
<td>-1.310E-04</td>
<td>9.851E-03</td>
</tr>
<tr>
<td>6.875</td>
<td>191</td>
<td>239</td>
<td>2.97</td>
<td>5.176E-04</td>
<td>9.028E-03</td>
</tr>
<tr>
<td>7.000</td>
<td>005-06</td>
<td>567-02</td>
<td>5.84</td>
<td>-3.901E-04</td>
<td>6.157E-03</td>
</tr>
<tr>
<td>7.125</td>
<td>191</td>
<td>144</td>
<td>8.52</td>
<td>-2.965E-04</td>
<td>3.481E-03</td>
</tr>
<tr>
<td>7.250</td>
<td>354</td>
<td>306</td>
<td>10.6</td>
<td>-2.137E-04</td>
<td>1.397E-03</td>
</tr>
<tr>
<td>7.375</td>
<td>462</td>
<td>419</td>
<td>11.7</td>
<td>-1.002E-04</td>
<td>2.611E-04</td>
</tr>
<tr>
<td>7.500</td>
<td>500</td>
<td>465</td>
<td>12.0</td>
<td>1.661E-06</td>
<td>4.468E-05</td>
</tr>
<tr>
<td>7.625</td>
<td>462</td>
<td>435</td>
<td>11.9</td>
<td>6.201E-05</td>
<td>8.331E-05</td>
</tr>
<tr>
<td>7.750</td>
<td>354</td>
<td>342</td>
<td>11.4</td>
<td>1.604E-04</td>
<td>5.908E-04</td>
</tr>
<tr>
<td>8.000</td>
<td>319-06</td>
<td>882-02</td>
<td>7.96</td>
<td>3.823E-04</td>
<td>4.043E-03</td>
</tr>
</tbody>
</table>

---

**Figure 2.** (Continued).
Figure 3. Execution of Interface Program.
3P

THE PERIOD OF THE TIME FUNCTION = 4.0000-06 SEC

ENTER 100 FOR PRINT, 810 FOR CALCOMP PLOT, OR 801 FOR TTY PLOT
100

<table>
<thead>
<tr>
<th>LINE</th>
<th>REAL</th>
<th>IMAG</th>
<th>DP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-2.7697-02</td>
<td>0.0000</td>
<td>-31.15</td>
</tr>
<tr>
<td>2</td>
<td>-4.9144-04</td>
<td>6.6766-04</td>
<td>-61.63</td>
</tr>
<tr>
<td>3</td>
<td>2.0001-02</td>
<td>7.9110-03</td>
<td>-33.31</td>
</tr>
<tr>
<td>4</td>
<td>-9.1365-05</td>
<td>-6.3436-04</td>
<td>-63.86</td>
</tr>
<tr>
<td>5</td>
<td>-2.3269-02</td>
<td>9.5285-03</td>
<td>-32.06</td>
</tr>
<tr>
<td>6</td>
<td>-6.6337-04</td>
<td>2.3214-04</td>
<td>-63.06</td>
</tr>
<tr>
<td>7</td>
<td>2.0155-02</td>
<td>-4.5856-02</td>
<td>-25.30</td>
</tr>
<tr>
<td>8</td>
<td>-1.9304-04</td>
<td>-1.1629-03</td>
<td>-58.57</td>
</tr>
<tr>
<td>9</td>
<td>-7.8478-02</td>
<td>-1.6430-03</td>
<td>-22.10</td>
</tr>
<tr>
<td>10</td>
<td>-5.3355-04</td>
<td>3.9614-04</td>
<td>-63.55</td>
</tr>
<tr>
<td>11</td>
<td>2.3991-01</td>
<td>1.2146-02</td>
<td>-12.72</td>
</tr>
<tr>
<td>12</td>
<td>1.0943-03</td>
<td>-1.6259-03</td>
<td>-54.17</td>
</tr>
<tr>
<td>13</td>
<td>-1.3978-01</td>
<td>3.1161-02</td>
<td>-17.43</td>
</tr>
<tr>
<td>14</td>
<td>-1.7313-03</td>
<td>-8.6959-05</td>
<td>-55.22</td>
</tr>
<tr>
<td>15</td>
<td>1.4733+00</td>
<td>-9.2625+00</td>
<td>4.63</td>
</tr>
<tr>
<td>16</td>
<td>-8.8963-04</td>
<td>-3.4519-03</td>
<td>-48.96</td>
</tr>
<tr>
<td>17</td>
<td>6.9555+00</td>
<td>0.0000</td>
<td>16.85</td>
</tr>
<tr>
<td>18</td>
<td>-8.9462-04</td>
<td>3.4519-03</td>
<td>-48.96</td>
</tr>
<tr>
<td>19</td>
<td>1.4733+00</td>
<td>2.2625+00</td>
<td>8.63</td>
</tr>
<tr>
<td>20</td>
<td>-1.7313-03</td>
<td>8.6959-05</td>
<td>-55.29</td>
</tr>
<tr>
<td>21</td>
<td>-1.3978-01</td>
<td>-3.1161-02</td>
<td>-17.43</td>
</tr>
<tr>
<td>22</td>
<td>1.0943-03</td>
<td>1.6259-03</td>
<td>-54.17</td>
</tr>
<tr>
<td>23</td>
<td>2.3991-01</td>
<td>-1.2146-02</td>
<td>-12.72</td>
</tr>
<tr>
<td>24</td>
<td>-5.3355-04</td>
<td>-3.9614-04</td>
<td>-63.55</td>
</tr>
<tr>
<td>25</td>
<td>-7.8478-02</td>
<td>1.6430-03</td>
<td>-22.10</td>
</tr>
<tr>
<td>26</td>
<td>-1.9304-04</td>
<td>1.1629-03</td>
<td>-58.57</td>
</tr>
<tr>
<td>27</td>
<td>2.0155-02</td>
<td>4.5856-02</td>
<td>-25.30</td>
</tr>
<tr>
<td>28</td>
<td>-6.6337-04</td>
<td>-2.3214-04</td>
<td>-63.06</td>
</tr>
<tr>
<td>29</td>
<td>-2.3269-02</td>
<td>-9.5285-03</td>
<td>-32.06</td>
</tr>
<tr>
<td>30</td>
<td>-9.1365-05</td>
<td>6.3436-04</td>
<td>-63.96</td>
</tr>
<tr>
<td>31</td>
<td>2.0001-02</td>
<td>-7.9110-03</td>
<td>-33.31</td>
</tr>
<tr>
<td>32</td>
<td>-4.9144-04</td>
<td>-6.6766-04</td>
<td>-61.63</td>
</tr>
</tbody>
</table>

ENTER 0 FOR PRINT OF DATA SET FROM CIRCUIT
ENTER PLUS DATA SET NUMBER FOR FIRST FCN
ENTER MINUS DATA SET NUMBER FOR TIME FCN

Figure 3. (Continued).
**Figure 3.** (Continued.)
after execution begins indicate a data set size of 65, the number of
points in a transform to be 32, and the number of data sets to be 5
(these correspond to the 5 variables in the PLOT statement, see Figure
2). A message set then follows which requests the entry of an integer,
a zero for print of a data set, a positive data set number for output
of the frequency function, or a negative data set number for output of
the time function.

The example shows the entry of a zero followed by another zero in
answer to the next question, thus indicating data set number 0 (the time
listing). This is followed by the 65 entries from CIRCUS making up data
set 0. After the printing of these data is complete, the original ques-
tions pertaining to the data desired are repeated as shown in the example
of Figure 3. This time a print of data set 3, the node 4 output volt-
age, from CIRCUS was requested.

The next response was a +3 requesting the frequency domain repre-
sentation of data set number 3. The program then requests the value of
ISTART, an integer specifying the starting index for the 32 values to be
used in the transform. Care should be exercised to insure that ISTART
is never greater than the number of points in the data set minus the num-er of points required for a transform; otherwise, the transform data
will be taken from two adjacent data sets (or partially from an adjacent
storage area) and the transform will be meaningless.

After entry of the value of ISTART, the period of the time function
is displayed. This display occurs only on the first call for either a
time function or a frequency function.

Next a statement relative to the type output desired is displayed.
The code for response to this statement is 100 for print, 010 for Cal-
comp plot, and 001 for teletype plot. The example shows a request for
print, followed by a printing of the 32 values of the frequency function
obtained from the original data set number 3 starting at array element
number 32. The three output types may be called individually, in pairs,
or all at one time. For example, responding with 111 would generate
printed output, a Calcomp plot, and a teletype plot of the specified data
set. The response 101 would produce printed output and a teletype plot.
After execution of the print command in the example of Figure 3, the program cycles back to the first set of questions and asks again for the entry of a 0, or plus or minus a data set number. The response in the example of -3 indicates a request for the time waveform of data set number 3. Next, in response to the questions generated by the program, ISTART is specified as 32, and an output type of 100 (printed output) is entered by entry of the number 100. This is followed by a print of the 32 data points which would be input to the FFT had a frequency function been called. These 32 points are from data set 3, starting at element number 32 and ending at element number 63. This provision for selecting any 32 consecutive points desired from a 65 point data set allows the first part of the data to be skipped when it contains a transient response.

After execution of the given command the program again responds with a request for output data type desired. The example of Figure 3 shows a request for the frequency function of data set 3 starting with data set element number 32 and for a teletype plot of these data by entry of 001 to the request for output type. The program next asks for entry of FLO, the lowest frequency desired in the plot. This can be any frequency compatible with the size of the stored frequency function. Entry of this frequency is followed by a request for entry of FHI, the highest frequency desired in the teletype plot. Entry of these two frequencies satisfies the requirements of the program and a plot is produced on the teletype as shown in the example of Figure 4. The ordinate of the plot is displayed horizontally and is calibrated in decibels with automatic scaling. The abscissa of the plot appears vertically (along the teletype paper) and can be any length. This axis along the paper is the frequency axis extending from the FLO to FHI specified.

Completion of the teletype plot again generates the request for the output data desired which in the example is answered by a -3, a request for the time function of data set number 3. Again the request for ISTART is responded to with a 32 and a teletype plot is requested by entry of 001. Since a time plot has been specified, the program requests that the starting and stopping value of the index of the data array be entered.
ENTER I START
30
THE PERIOD OF THE TIME FUNCTION = 4.0000-06 SEC

ENTER I0 FOR PRINT, I10 FOR CALCUMP PLOT, 0P 0P1 FOR TTY PLOT
0P1
ENTER FLO
3.75E6
ENTER FHI
3.75E6
NSIZE = 31
FREQUENCY (MHZ) DECIBELS
-30 -20 -10 0 10 20
--- --- --- --- --- ---
--- --- --- --- --- ---
-3.750 I
-3.500 I
-3.250 I
-3.000 I
-2.750 I
-2.500 I----
-2.250 I
-2.000 I-------
-1.750 I
-1.500 I-----------------
-1.250 I
-1.000 I-----------------
-0.750 I
-0.500 I------------------------------
-0.250 I
-0.000 I-----------------------
+0.250 I
+0.500 I-----------------------
+0.750 I
+1.000 I-----------------
+1.250 I
+1.500 I-----------------
+1.750 I
+2.000 I-------
+2.250 I
+2.500 I----
+2.750 I
+3.000 I
+3.250 I
+3.500 I
+3.750 I
FREQ
ENTER 0 FOR PRINT OF DATA SET FROM CIRCUS
ENTER PLUS DATA SET NUMBER FOR FREQ FCN
ENTER MINUS DATA SET NUMBER FOR TIME FCN
-3

Figure 4. Spectrum of Data Set 3.

18
The data array available for the time plot is contained in an array having the length of the transform array specified (in this case 32). Any stopping value up to the maximum length of the array may be specified (in this case 32). The entry shown is 32. The next request by the program is that for NJUMP which specifies the number of points skipped in the array between points plotted. This allows skipping points between plotted points and is a very useful feature when large data arrays are processed. In the example shown the transform data array size is small and the value of NJUMP is entered as 1. The output is shown in Figure 5 which displays the time waveform of the amplifier output.

The last request illustrated is that for a Calcomp plot. The Calcomp plotting routines and methods used for setting up the plot files are probably unique with the Georgia Tech 1108 computer. Similar routines and specific control directives should allow the use of the plotting routines with a minimum of changes at other installations. The Calcomp frequency plot routine contains as its last instruction a write statement to write a message "Plot Complete" indicating that the program has processed the plot routine. The command to generate a Calcomp plot of the frequency function is a plus data set number (3 in this case) followed by ISTART (32) followed by 010. These instructions are shown in Figure 6, and the resulting plot in Figure 7.

To exit the program, the command @EOF is given. The remaining information relative to the Calcomp plot and the time to execute the run is generated by the computer.

B. Use of ECAP for Transfer Function Evaluation

One of the difficulties frequently encountered in the use of CIRCUS was that of obtaining an accurate representation of a circuit for analysis. If the circuit description presented to CIRCUS is not an accurate representation of the actual circuit, then the results produced by CIRCUS will not accurately represent the circuit response. An example of this type of difficulty is that of entering an interstage bandpass filter which contains mutual inductance and is situated between two active
Figure 5. Time Waveform of Data Set 3.
ENTER ISTART

32  
THE PERIOD OF THE TIME FUNCTION = 4.0000E-06 SEC  

ENTER 100 FOR PRINT, 010 FOR CALCOMP PLOT, OR 001 FOR TTY PLOT  
010  
ENTER THE HIGHEST DESIRED FREQ IN THE SPECTRUM, fMAX  
3.75E6  

PLOT COMPLETED  
ENTER A FOR PRINT OF DATA SET FROM CIRCUS  
ENTER PLUS DATA SET NUMBER FOR FREQ FCN  
ENTER MINUS DATA SET NUMBER FOR TIME FCN  
00F  

PLOT  2.7 MIN  1.3 FT  00F  112072191104  
END  1348 MLSEC  

Figure 6. Request for Calcomp Plot.
Figure 7. Calcump Plot of the Frequency Spectrum of the Time Waveform shown in Figure 5.
devices. Such filters usually have bandwidths wide enough to pass the significant sidebands of a signal of interest after allowing for such factors as frequency instability and temperature effects. For CIRCUS to provide an accurate representation of the circuit response, the transfer function of the filter must meet these bandwidth requirements. Since CIRCUS determines the circuit response by solving for its time response (which yields only a time waveform), the transfer characteristics of the bandpass filter cannot be readily determined from CIRCUS calculations.

In a search for better methods of coping with such problems, other analysis programs were investigated. Those having an a-c analysis capability in the frequency domain were of particular interest since the circuit description could be entered in a manner similar to that used in CIRCUS. If the computed circuit response was not as desired, adjustments in circuit parameters could be made and the transfer function evaluated again. Thus the variation of the transfer function with variation in the value of circuit parameters could be obtained.

With such an a-c analysis capability, much of the guesswork of entering a circuit into CIRCUS could be removed. This is not to say that the response of a filter, for example, is independent of the devices preceding and following it, but that values of circuit parameters could be determined to yield the desired filter response. These could then be adjusted for drive and load impedances, possibly by using CIRCUS to calculate the impedances presented by the circuit external to the filter.

The analysis program ECAP was selected to be used for the a-c analysis calculations. This selection was based largely on the availability of ECAP on the Georgia Tech Univac-1108 computer.

To illustrate the use of ECAP, a fairly complex circuit with a known response was selected. This circuit was the CCG telemetry bandpass filter for which calculated and experimentally determined responses have been previously obtained [1]. A circuit diagram of this filter is shown in Figure 8 with the input transformer replaced by its "T" equivalent circuit. The underlined numbers in the figure show the node numbers used for entry of the circuit into ECAP. The branches are shown in the
Figure 8. Circuit Diagram of CCS Telemetry Filter.
figure as numbers preceded by a "B" and are from left to right and from top to bottom. The circuit description in the format required by ECAP is shown in Figure 9. The circuit response calculated by ECAP agreed very closely with that obtained earlier, and is shown in Figure 10 by the "x" points superimposed on the response calculated previously [1].

A sample of the output produced by ECAP is shown in Figure 11. The output of ECAP is not easily interpreted. ECAP essentially re-executes for each new frequency and therefore produces an excessive amount of superfluous output information. A much easier to interpret output format would result if the output data were displayed in tabular form.

The ability of ECAP to solve for a transfer function of a circuit has been demonstrated and a familiarity with the user program interface was obtained.

C. Conclusions and Recommendations

An interface with the time domain program CIRCUS has been constructed which allows the frequency domain representation of time waveforms generated with CIRCUS to be produced. These frequency domain representations of signals are generated by use of the FFT algorithm. In addition, the capability has been provided for easily obtaining listings of the data sets obtained from CIRCUS, or for generating Calcomp or teletype, time or frequency plots of portions of these data sets.

The a-c analysis capabilities of ECAP have been investigated and found to be a useful tool for accurate preparation of data for input to programs such as CIRCUS. This a-c analysis capability is particularly useful in the evaluation of transfer functions.

It is recommended that the capabilities of the two programs, CIRCUS and ECAP (preferably the latest versions of these programs) be interfaced such that nonlinear circuits could be handled with CIRCUS and linear circuits with ECAP. The link between the frequency and time domains would be the fast Fourier transform algorithm. Part of this interface has already been established with techniques discussed in this section. There remains a need to interface the a-c analysis capabilities of ECAP with the frequency domain data obtained from CIRCUS. A study should be undertaken to determine an efficient method of providing this interface.
AC ANALYSIS

B1  N(0,1), R = 50., E = 1./0.
B2  N(1,2), L = -5.358E-6
B3  N(0,2), L = 6.06E-6
B4  N(2,3), L = 51.94E-6
B5  N(3,4), R = 7.76
B6  N(0,4), C = 20.7E-12
B7  N(0,4), R = 5.1E3
B8  N(4,5), C = 480.7E-12
B9  N(5,6), R = 1.7
B10 N(6,7), L = 10.8E-6
B11 N(0,7), C = 1368.7E-12
B12 N(5,8), C = 510.7E-12
B13 N(8,9), R = 8.01
B14 N(9,10), L = 51.7E-6
B15 N(8,10), C = 16.1E-12
B16 N(10,11), R = 2.51
B17 N(11,12), L = 16.7E-6
B18 N(0,12), C = 2200.7E-12
B19 N(10,12), C = 344.7E-12
B20 N(10,13), R = 8.01
B21 N(13,14), L = 51.7E-6
B22 N(10,14), C = 16.9E-12
B23 N(0,14), R = 4.7E3
B24 N(4,15), C = 570.7E-12
B25 N(0,15), C = 1800.7E-12
B26 N(0,15), R = 620.

FIREQUF./qCU = 1024.7E3
PRINT, VOLTAGES
MODIFY
FREQUENCY = .5E6 (+25)3*E6
EXECUTE
END

Figure 9. Circuit Description in ECAP Format.
<table>
<thead>
<tr>
<th>NODES</th>
<th>NODE VOLTAGES</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-4</td>
<td>56120345-01</td>
</tr>
<tr>
<td></td>
<td>84115222+02</td>
</tr>
<tr>
<td>5-8</td>
<td>89344465-01</td>
</tr>
<tr>
<td></td>
<td>63921702+02</td>
</tr>
<tr>
<td>9-12</td>
<td>13643413-01</td>
</tr>
<tr>
<td></td>
<td>-10128080+03</td>
</tr>
<tr>
<td>13-15</td>
<td>12923102-01</td>
</tr>
<tr>
<td></td>
<td>85511819+02</td>
</tr>
</tbody>
</table>

**Figure 11.** Sample of ECAP Output.
III. MATHEMATICAL MODELING OF COMMUNICATION SYSTEMS
ON A BLOCK BASIS

A. General Description

The Frequency and Time Circuit Analysis Technique (FATCAT) is a computer implemented program for analyzing communications circuits and is designed on a block modeling approach. Circuits are represented as a linear collection of sub-assemblies and FATCAT provides a model block for a variety of sub-assemblies including signal sources, filters, demodu-


lators, amplifiers, limiters, mixers, etc. The program is designed for use with either remote terminal or batch processing mode. For remote terminal operation the program is conversational in nature and provides considerable flexibility to the user. Input statements to specify the circuit configuration, to direct processing, and to direct output are given in alphanumeric codes along with numeric specification of param-


eters. The input formats were designed to be relatively simple and easy to use.

FATCAT is designed for steady-state analysis of circuits in which the signal flow is sequential; no provision is presently included for feedback loops. The model is designed to operate in both the time do-


main and the frequency domain. The signal being processed is stored in a complex array as either a frequency domain representation of the sig-


nal or one complete period of a time wave form. Transition from one state to another is made using a subroutine which performs a fast Fourier transform. Each model block was developed in whichever domain was most convenient for modeling that block. When calling for signal processing through any block, the domain representation of the signal which current-


ly exists is checked and, if necessary, the conversion is automatically made. The same automatic conversion is available on output calls; this permits examining either the time waveform or the frequency spectrum at the output of any block. Outputs include both printed and plotted values of the time waveform or the frequency spectrum.

The entire program was developed and implemented on a Univac-1108 at Georgia Tech. Operation and command structure of the program along with
examples of using the program are given in the following sections. A brief description of each software unit along with a program listing is given in Appendix C.

After development of the program was completed to its present form, suitable modifications were made to implement it on a SIGMA-5 computer at Marshall Space Flight Center. Appendix D gives program listings of those routines that were changed for use on the SIGMA-5.

B. Program Control

Operation of the program is accomplished by giving a sequence of defined commands, each command being given basically with an alphanumeric string of 6 or less characters. In most cases other data, usually numeric will follow the basic command.

The set of commands can be broken into several categories (1) input (block specification) commands, (2) control commands, and (3) special commands.

After starting execution of the program, the first input command will either be specification of the first block of the circuit, or one of two special commands. These two special commands are (1) the ability to list the command structure of the program, and (2) the ability to list the input formats for all input commands. These two commands are provided as a convenience to the remote terminal user and are described in detail in Section IIIB3.

1. Block Input Commands

Each circuit block is specified with a line entry composed of the alphanumeric code for that block, followed by the parameters of the block. The entry items on a line are separated by commas; no comma is used after the last item. Each block has a specific entry format and these are described below.

The blocks of a circuit are entered sequentially in the order that they occur in the circuit. Thereafter they are identified by number, i.e., the first block entered is block number 1, the second is block
number 2, etc. The block numbers are important in directing processing and outputs.

Flexibility is provided for the remote terminal user in entering the blocks. All blocks comprising a circuit may be entered before any processing takes place, or only part of the blocks may be entered and processed through before the remaining blocks are entered. The user has the option of adding new blocks to the end of the input string at any time (provided the block count does not exceed the maximum allowable number of blocks, currently set at 20). One additional restriction is that if a block containing a source frequency (a mixer with a local oscillator, for example) is to be added after processing has begun, the source frequency must be chosen to be periodic in the time interval represented by the time function stored in the data array. This time interval (PERIOD) is printed out at the beginning of processing so that the choice of a suitable source frequency should be relatively easy.

a. Sources

Two signal source blocks are provided. The first simulates a signal generator and is called with

SIGGEN, FO, FMOD, AM, PM, FM, A

where

FO = carrier frequency (Hz),
FMOD = modulation frequency (Hz),
AM = percent AM modulation,
PM = peak phase deviation (radians),
FM = peak frequency deviation (Hz), and
A = peak amplitude (volts).

The signal generator block provides only sine wave modulation. Combinations of AM-PM or AM-FM modulation may be specified, but not PM-FM (either PM or FM must be zero). As an example, a signal generator with a carrier frequency of 1.0 MHz, a modulation frequency of 10.0 kHz, 50% AM modulation plus FM modulation with β = 2, and a peak amplitude of one volt would be entered as SIGGEN, 1.E6, 1.E4, 50., 0., 2.E4, 1.
A call for processing this block will generate the appropriate time waveform and store it in the data array.

The second signal source is a flat spectrum (impulse time function) generator specified by

\( \text{FLATSP, AMP, DELF, N} \)

where

- \( \text{AMP} \) = amplitude of spectrum lines,
- \( \text{DELF} \) = frequency separation of spectral lines (Hz), and
- \( \text{N} \) = array size.

The user specified values of \( \text{DELF} \) and \( \text{N} \) determine the overall frequency spread of the spectrum being produced and hence determines the period of the impulse function being represented. When using \( \text{FLATSP} \), the usual computation of the array size and period is not used. The actual value of \( \text{N} \) that is used will be a power of 2; if the input \( \text{N} \) is not a power of 2, it will be automatically raised to meet this requirement. (Example: if \( \text{N} = 500 \) is specified, \( \text{N} = 512 \) will be used.)

b. Filters

Eleven different filter blocks are included which simulate the action of Butterworth, Tchebysheff, and synchronously tuned filters in low pass, high pass, bandpass, and (except for sync tuned) band stop configurations. Since the input statements are similar, the commands will be grouped by types and the inputs for Butterworth, Tchebysheff, and synchronously tuned filters of each type given in that order. The commands for low pass filters are:

- \( \text{BWLOWP, FC, NR} \)
- \( \text{CHLOWP, FC, NR, EPSDB} \)
- \( \text{SYNLP, FC, NR} \)

where

- \( \text{FC} \) = corner frequency (Hz),
- \( \text{NR} \) = number of filter sections, and
- \( \text{EPSDB} \) = Tchebysheff ripple factor in decibels.
The inputs for high pass filters are:

BWHIP, FC, NR
CHHIP, FC, NR, EPSDB
SYNHP, FC, NR

where the parameters are identical to those defined for low pass filters.

The inputs for bandpass filters are:

BWBNDP, FO, BW, NR
CHBNDP, FO, BW, NR, EPSDB
SYNBP, FO, BW, NR

where

FO = center frequency (Hz),
BW = total bandwidth (Hz), (3 dB bandwidth for synchronous and
Butterworth filters and ripple amplitude bandwidth for
Tchebyscheff filters),
NR = number of filter sections, and
EPSDB = Tchebysheff ripple factor in decibels.

The inputs for band stop filters are:

BWBSTP, FO, BW, NR
CHBSTP, FO, BW, NR, EPSDB

and the parameters are identical to those for band pass filters.

c. Demodulators

Demodulators for AM, FM, and PM are specified, respectively

with

AMDEMO, FO
FMDEMO, FO
PHDEMO, FO

where

FO = center frequency (Hz).

d. Other Blocks

In addition to the blocks categorized above, several other
blocks are provided. An amplifier is specified with

\[ \text{AMP, Gain} \]

where

\[ \text{Gain} = \text{amplifier gain in dB.} \]

A limiter is specified with

\[ \text{LIM, CL, CH, GL} \]

where

\[ \text{CL} = \text{low clipping level (volts),} \]
\[ \text{CH} = \text{high clipping level (volts), and} \]
\[ \text{GL} = \text{limiter gain (volts/volt).} \]

A frequency multiplier (a wide band harmonic generator) is specified with

\[ \text{FRQMUL} \]

(no parameters necessary).

An ideal multiplier can be specified with

\[ \text{IDLMUL, ALO, FLO} \]

where

\[ \text{ALO} = \text{amplitude of the LO signal (volts), and} \]
\[ \text{FLO} = \text{frequency of LO signal (Hz).} \]

2. Control Commands

A number of control commands are used to direct processing and to generate output. Processing of the signal to the output of any block is initiated with the command

\[ \text{BLOCK, N} \]

where \( N \) is the number of the block. Note that processing occurs only when this command is given; entry of a block specification does not cause processing through that block.

Signal processing is non-reversible; if processing to the output of a given block has been completed, the outputs of earlier blocks are no longer available. Error detecting features are included in the program so that a BLOCK call with an \( N \) that is wrong will not upset the computations.
The command

PRIME FACTORS

will list the prime factors of all source frequencies. These can be helpful when it is necessary to adjust one or more frequencies slightly so that all source frequencies will be periodic on an interval of reasonable size. A periodic interval that is too long will require an array of excessive size.

The command

END OF JOB

will terminate a run.

Other control commands are used to generate output. Outputs include printed listing, printer type plots, and plotting equipment outputs of both the time function and the frequency function. When any of the output commands are given, the output data is from the block output where processing currently stands. For example, in a six-block circuit, if processing through block 3 has been effected by the command BLOCK 3, then any output generated will be for the output of block 3. To examine the signal at the output of block 5, it will first be necessary to process the signal with BLOCK, 5, and then call for the output.

For a printed listing of the time function, the command is

PRINT T, NSTART, NSTOP

where

NSTART = the starting array index, and
NSTOP = the final array index (not to exceed the array size being used).

In conversational mode, an alternative command is

PRINT

following which the machine will ask for the starting and stopping indices.

*Only the first six characters of any command are important, and spaces (blanks) are ignored. This command can be abbreviated to PRIMEF.
For a printed listing of the frequency function the command is

\[ \text{PRINTF, FLO, FHI} \]

where

- \( \text{FLO} = \) low frequency limit (Hz), and
- \( \text{FHI} = \) high frequency limit (Hz).

Alternatively, the command

\[ \text{PRINTF} \]

will produce questions asking for FLO and FHI, following which the same output will be generated. The use of these parameters permits only the portion of the spectrum of interest to be printed.

The commands for plotted output utilize the same parameter forms described above, and when operating from a remote terminal the commands can be given with or without the parameters. When the parameters are omitted from the command, the computer will ask for them.

Printer plots (line printer for batch processing, teletype plots for operation from a remote teletype terminal) of the time function are obtained with

\[ \text{TPLOTT, NSTART, NSTOP, NJUMP} \]

where

- \( \text{NJUMP} = \) interval between plotted points and the other parameters are the same as for PRINTT.

Printer plots of the frequency spectrum are generated by

\[ \text{TPLOTF, FLO, FHI} \]

where FLO and FHI are the same as for PRINTF.

Plotter outputs are similarly generated with

\[ \text{CPLOTT, NSTART, NSTOP} \]

for the time function, and

\[ \text{CPLOTF, FLO, FHI} \]

for the frequency function.
3. Special Commands

Three special commands are included for use in connection with establishing the size of the data array. Initial processing of any circuit containing source frequencies (such as those originating in a signal generator or the local oscillator of a mixer) will start with a determination of the smallest time increment on which all of the source frequencies are periodic, and a calculation of an array (sample) size that meets the Nyquist criterion for all frequencies. The calculated period and array size will then be printed out and the user asked if the size is satisfactory. At this point the program is positioned at a special input position, at which only four commands will be recognized:

YES
NO
N, NSIZE
END OF JOB

where NSIZE is an integer which specifies the size of the array. If the answer is NO, the program will ask for input of an integer for NSIZE. For either method of entering NSIZE, the input integer will be adjusted to a power of 2 by increasing it if necessary, and the new values of N, IGAM, and DELTA-T will be printed out. The question of whether these values are satisfactory will then be repeated. Thus this section of the program is a loop and will be exited only when a YES response (or an END OF JOB) is given. Processing of the signal to the block output designated will continue after a YES is entered provided N lies within acceptable limits. To be acceptable, N must be at least as large as the value initially computed in order to meet the Nyquist criterion; at the same time, it must be no larger than the size of the main data array declared in the main program. Since cases can occur where both these conditions cannot be met, recognition of END OF JOB has been included at this point to permit a normal termination of the run. Any attempt to continue a run with N outside the required limits will cause an error termination.

Two other special commands are included as an aid to the remote
terminal user. The command

LIST COMMANDS

followed by a comma and one of several second words will produce a list-
ing of part or all of the FATCAT commands. Permissible second words, and
their effect are:

LIST COMMANDS - print instructions for using list commands,
ALL - list all operating commands,
SOURCES - list commands for inserting source blocks,
FILTERS - list commands for inserting filters,
DEMODULATORS - list commands for inserting demodulators,
MISC - list other block input commands, and
CONTROL COMMANDS - list control commands.

The above instruction will only list the command word for each in-
struction along with identifying information. Input data formats will
not be listed. If the user wants information on the input data format
for any block specification command, the command

INPUT FORMAT

followed by a comma and the name of any block specification command will
produce a listing of the complete command with the required parameters,
and will identify the meaning of each parameter. For example, the com-
mand

INPUT FORMAT, BWBNBP

will list the complete input instructions for a Butterworth band pass
filter.

The command

INPUT FORMAT, ALL

will list the formats for all of the block specification commands.

*May be abbreviated to LISTCO.
C. Simulation Examples using FATCAT

As examples of the use of the FATCAT program, two sample problems will be presented. Both examples represent receiving systems, the first being an amplitude modulation receiver and the second a frequency modulated receiver.

1. AM Receiver

A block diagram of an amplitude modulated superheterodyne receiver is shown in Figure 12. Figure 12a shows the actual configuration of the receiver, while Figure 12b shows the way that the receiver would presently be modeled using FATCAT. The signal source is presently represented by a signal generator which produces single tone modulated radio frequency signals. In the final version of FATCAT as envisioned, this signal source could be modeled as an actual transmitter with modulators, frequency multipliers and amplifiers. FATCAT does not presently include a propagation model or antenna models. Therefore the output of the signal generator represents the input signal to the receiver. The control program is structured so that additional model blocks such as those for transmitters, propagation, antennas, and many more, can be easily added to the model blocks already included in the program.

The first step in the analysis of the amplitude modulated receiver is a command for the execution of the FATCAT program. Figure 13 shows the command for execution of the program contained in file "R" as absolute element "U". Following this command the program responds with the word "START", which indicates that the program is ready for input commands. Following the sequence of blocks shown in Figure 12b the first input instruction given is that for an AM signal generator with a carrier frequency of 10 MHz, a modulation frequency of 10 kHz, 50 percent amplitude modulation, no phase or frequency modulation, and a carrier peak amplitude of 1 microvolt. This is the "SIGGEN" command shown in Figure 13. Next the command, "BLOCK, 1" is given which requests the program to process the submitted instruction and arrive at the output of block 1. After the first command to process to the output of a block for which
Figure 12. Block Diagram of Amplitude Modulation Receiver and FATGAT Model of Receiver.
Figure 13. Start of FATCAT Run for AM Receiver, Showing Spectrum at the Output of the Signal Generator.
input data has been submitted (which can be a single block or multiple blocks) the output information shown in Figure 13 is given. This information contains PERIOD which gives the period of the time function, DELTA-F, which is the spacing of possible spectral component in the frequency domain, N which is the array size needed to meet the minimum Nyquist requirements, IGAM the exponent to which 2 is raised to give N, and DELTA-T the sampling interval in the time domain. After the quantities have been printed the program asks if these values are satisfactory. Response to this question can be "YES" or "NO". If a "YES" is given processing proceeds to the output of the block specified. In the example of Figure 2 processing is completed through block 1. If "NO" had been given in response to the above question the program would have responded with "ENTER N, VALUE" which allows a new value of N to be inserted. This provision allows the number of samples per cycle of the highest frequency in the input signals to be increased. This is especially useful for plotting since just over two samples per cycle, which meets the Nyquist criterion, is not adequate for detailed plotting of the time function.

Next it is desirable to observe the output of the signal generator, and an observation of the frequency spectrum was selected. This is produced by the command "TPlotF" (teletype plot of the frequency function) which is followed by a request for the low and high frequency limits of the spectrum to be observed. These were specified to be 9.9 MHz to 10.1 MHz as shown in Figure 13 and the program produced the spectrum plot shown in that figure. Note that the ordinate of the plot is automatically scaled. The spectrum of the AM signal is displayed with a carrier amplitude of 0.5 microvolts or -126 dB (one side of a two-sided spectrum); the amplitudes of each sideband component is 0.125 microvolt or -138 dB. A print of the frequency function is shown in Figure 14 which shows the amplitude and phase of these spectral components.

The next block in the diagram of Figure 12b is that of a single section Butterworth bandpass filter with a total 3 dB bandwidth of 20 kHz. This block command along with the command to process to the output of block 2 is shown at the top of Figure 15. The 3 dB point of the filter was placed at the sideband frequencies of the AM signal so that the effect of the filter could be observed. The print of Figure 15 when compared
```
PRINTF
PRINTF
ENTER LOW, HIGH FREQUENCIES
9.9E6, 10.1E6

<table>
<thead>
<tr>
<th>LINE</th>
<th>FREQ</th>
<th>REAL</th>
<th>IMAG</th>
<th>MAG</th>
<th>DP</th>
<th>PHASE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015</td>
<td>9.9E+06</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>-236.18</td>
<td>87.4</td>
</tr>
<tr>
<td>2016</td>
<td>9.91E+06</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-234.92</td>
<td>92.1</td>
</tr>
<tr>
<td>2017</td>
<td>9.92E+06</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>-234.62</td>
<td>84.1</td>
</tr>
<tr>
<td>2018</td>
<td>9.93E+06</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>-233.22</td>
<td>82.6</td>
</tr>
<tr>
<td>2019</td>
<td>9.94E+06</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-232.73</td>
<td>96.0</td>
</tr>
<tr>
<td>2020</td>
<td>9.95E+06</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>-230.36</td>
<td>88.2</td>
</tr>
<tr>
<td>2021</td>
<td>9.96E+06</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>-228.22</td>
<td>89.0</td>
</tr>
<tr>
<td>2022</td>
<td>9.97E+06</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-227.58</td>
<td>92.0</td>
</tr>
<tr>
<td>2023</td>
<td>9.98E+06</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>-221.96</td>
<td>82.1</td>
</tr>
<tr>
<td>2024</td>
<td>9.99E+06</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-134.06</td>
<td>-98.0</td>
</tr>
<tr>
<td>2025</td>
<td>1.00E+07</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-126.92</td>
<td>-90.4</td>
</tr>
<tr>
<td>2026</td>
<td>1.01E+07</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-138.06</td>
<td>-98.0</td>
</tr>
<tr>
<td>2027</td>
<td>1.02E+07</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-222.49</td>
<td>-81.4</td>
</tr>
<tr>
<td>2028</td>
<td>1.03E+07</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-228.22</td>
<td>-89.2</td>
</tr>
<tr>
<td>2029</td>
<td>1.04E+07</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-229.39</td>
<td>-88.6</td>
</tr>
<tr>
<td>2030</td>
<td>1.05E+07</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-231.95</td>
<td>-88.3</td>
</tr>
<tr>
<td>2031</td>
<td>1.06E+07</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-235.10</td>
<td>-84.4</td>
</tr>
<tr>
<td>2032</td>
<td>1.07E+07</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-235.44</td>
<td>-78.3</td>
</tr>
<tr>
<td>2033</td>
<td>1.08E+07</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-237.59</td>
<td>-80.1</td>
</tr>
<tr>
<td>2034</td>
<td>1.09E+07</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-238.36</td>
<td>-96.3</td>
</tr>
<tr>
<td>2035</td>
<td>1.10E+07</td>
<td>- .000</td>
<td>.000</td>
<td>.000</td>
<td>-239.76</td>
<td>-86.4</td>
</tr>
</tbody>
</table>
```

Figure 14. Listing of the Frequency Function at the Output of the Signal Generator.
Figure 15. Listing of Frequency Function at the Output of the First Bandpass Filter.
with Figure 14 verifies this. The sidebands at 9.99 MHz and at 10.01 MHz have an amplitude of -141.07 dB out of the filter and had an amplitude of -138.06 at the input to the filter -- a difference of 3.01 dB. Note that the carrier amplitude is unchanged. A plot of the spectrum at the output of the filter is shown in Figure 16.

Next in the block diagram of the receiver is a mixer. This is modeled in FATCAT as an ideal mixer which produces sum and difference frequencies only. Another useful feature of FATCAT is the input format request command, shown at the top of Figure 17, which can be used when the input format of a block is not known. The request "INPUT FORMAT", followed by a comma, and then followed by the name of a block in the model library, yields the input parameters and their definition as shown in Figure 17 (use of this command has no effect on the circuit being processed). The ideal multiplier IDLMUL requires the peak amplitude of the local oscillator be entered as well as the frequency of the local oscillator. This amplitude in the example was given as 1 volt and the frequency as 9 MHz. This should result in the spectrum of the AM signal being shifted to 1 MHz and 19 MHz. The spectrum around the difference frequency is shown in Figure 17.

The next two blocks of the receiver are made up of a four section Butterworth bandpass filter with a center frequency of 1 MHz and a bandwidth of 20 kHz followed by an amplifier having a gain of 120 dB. The input commands for these blocks and the spectrum at the output of each block are shown in Figures 18 and 19, respectively.

Proceeding along the block diagram of Figure 12b, the next block is an amplitude demodulator. The input commands for this block and the output spectrum are shown in Figure 20. Note that the only two components in the spectrum displayed are the d-c component and the demodulated signal at 10 kHz.

Figure 21 shows the result of a call on the teletype time plot routine. This routine was called to plot the time function at the output of the demodulator. The request for the entry of NSTART, NSTOP, and NJUMP allows the starting point in the array containing the time function to be specified, the stopping point to be specified, and the number of points
**Figure 16. Spectrum at the Output of the First Bandpass Filter.**

TPLOTE
TPLOTE
ENTER LOW, HIGH FREQUENCIES
9.926, 10.156

N SIZE = 21

<table>
<thead>
<tr>
<th>FREQUENCY (MHz)</th>
<th>DECIBELS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-170</td>
</tr>
<tr>
<td>9.900</td>
<td>I</td>
</tr>
<tr>
<td>9.910</td>
<td>I</td>
</tr>
<tr>
<td>9.920</td>
<td>I</td>
</tr>
<tr>
<td>9.930</td>
<td>I</td>
</tr>
<tr>
<td>9.940</td>
<td>I</td>
</tr>
<tr>
<td>9.950</td>
<td>I</td>
</tr>
<tr>
<td>9.960</td>
<td>I</td>
</tr>
<tr>
<td>9.970</td>
<td>I</td>
</tr>
<tr>
<td>9.980</td>
<td>I</td>
</tr>
<tr>
<td>9.990</td>
<td>I</td>
</tr>
<tr>
<td>10.000</td>
<td>I</td>
</tr>
<tr>
<td>10.010</td>
<td>I</td>
</tr>
<tr>
<td>10.020</td>
<td>I</td>
</tr>
<tr>
<td>10.030</td>
<td>I</td>
</tr>
<tr>
<td>10.040</td>
<td>I</td>
</tr>
<tr>
<td>10.050</td>
<td>I</td>
</tr>
<tr>
<td>10.060</td>
<td>I</td>
</tr>
<tr>
<td>10.070</td>
<td>I</td>
</tr>
<tr>
<td>10.080</td>
<td>I</td>
</tr>
<tr>
<td>10.090</td>
<td>I</td>
</tr>
<tr>
<td>10.100</td>
<td>I</td>
</tr>
</tbody>
</table>
Figure 17. Spectrum at the Output of the Ideal Multiplier.
Figure 18. Spectrum at the Output of the Second Bandpass Filter.
AMP, 120.
AMP, 120.
BLOCK, 5
BLOCK, 5
PROCESSING COMPLETE THRU BLOCK 5
TPLOTF
TPLOTF
ENTER LOW, HIGH FREQUENCIES
9E6, 1.1E6

SIZE = 21

<table>
<thead>
<tr>
<th>FREQUENCY (MHz)</th>
<th>DECIBELS</th>
</tr>
</thead>
<tbody>
<tr>
<td>-60</td>
<td>-50</td>
</tr>
<tr>
<td>-40</td>
<td>-30</td>
</tr>
<tr>
<td>-20</td>
<td>-10</td>
</tr>
</tbody>
</table>

| 0.900 I         |
| 0.910 I         |
| 0.920 I         |
| 0.930 I         |
| 0.940 I         |
| 0.950 I         |
| 0.960 I         |
| 0.970 I         |
| 0.980 I         |
| 0.990 I         |
| 1.000 I         |
| 1.010 I         |
| 1.020 I         |
| 1.030 I         |
| 1.040 I         |
| 1.050 I         |
| 1.060 I         |
| 1.070 I         |
| 1.080 I         |
| 1.090 I         |
| 1.100 I         |
| FREQ           |

Figure 19. Spectrum at the Output of the Amplifier.
AM DEMO
AM DEMO
BLOCK, 6
BLOCK, 6
PROCESSING COMPLETE THRU BLOCK 6
TPLOTF
TPLOTF
ENTER LOW, HIGH FREQUENCIES
N = 200.43

NSIZE = 21

FREQUENCY (KHZ)                DECIBELS

-60  -50  -40  -30  -20  -10
  I--------I--------I--------I--------I--------I
           I
  0.000 I
  10.000 I
  20.000 I
  30.000 I
  40.000 I
  50.000 I
  60.000 I
  70.000 I
  80.000 I
  90.000 I
 100.000 I
 110.000 I
 120.000 I
 130.000 I
 140.000 I
 150.000 I
 160.000 I
 170.000 I
 180.000 I
 190.000 I
 200.000 I

Figure 20. Spectrum at the Output of the AM Demodulator.
Figure 21. Time Waveform at the Output of the AM Demodulator.
to be skipped between plotted points to be specified. For the plot of Figure 21 the starting value was 1, the stopping value was 2048, and 64 points were skipped between points plotted.

Figure 22 illustrates the capability of printing the time function by use of the command "PRINTT". Following this command a request for the entry of the low and high array indices is printed. Figure 22 shows a print of the time function from array index 1 to index 30.

A plot of the baseband spectrum at the output of the demodulator is shown in Figure 23 to demonstrate the flexibility of the plotting program. The spectrum can be seen to contain a positive and a negative frequency component as well as a d-c term.

Input of the final block in the AM receiver is shown in Figure 24. This is a five section Tchebysheff lowpass filter with a corner frequency of 10 kHz and an inband ripple of 1 dB. Observation of the 10 kHz component in Figures 23 and 24 indicates that it has been reduced on the order of 1 dB as it should since this component lies at the filter corner frequency.

Exit from the program is effected by entering the command END OF JOB, which is followed in Figure 24 by the time required to simulate the receiver.

2. FM Receiver

The second example is that of the analysis of a proposed frequency modulated receiver. The block diagram used for the FATCAT analysis of the receiver is shown in Figure 25.

The characteristics of the blocks making up the FM receiver are:

(1) A signal generator with a carrier frequency of 10 MHz, a modulation frequency of 10 kHz, no AM or PM modulation, FM modulation producing a peak frequency deviation of 30 kHz, and a carrier peak amplitude of 1 microvolt.

(2) A single section Butterworth bandpass filter with a bandwidth of 80 kHz.

(3) An ideal mixer with a local oscillator amplitude of 1 volt peak and a frequency of 9 MHz.
PRINT
PRINT
ENTER LOW, HIGH INDICES
1, 30

<table>
<thead>
<tr>
<th>LINE</th>
<th>REAL</th>
<th>IMAG</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.058-01</td>
<td>0.000</td>
</tr>
<tr>
<td>2</td>
<td>2.057-01</td>
<td>3.241-09</td>
</tr>
<tr>
<td>3</td>
<td>2.055-01</td>
<td>3.147-09</td>
</tr>
<tr>
<td>4</td>
<td>2.054-01</td>
<td>2.024-09</td>
</tr>
<tr>
<td>5</td>
<td>2.053-01</td>
<td>1.673-09</td>
</tr>
<tr>
<td>6</td>
<td>2.051-01</td>
<td>2.161-09</td>
</tr>
<tr>
<td>7</td>
<td>2.050-01</td>
<td>2.005-09</td>
</tr>
<tr>
<td>8</td>
<td>2.049-01</td>
<td>2.001-09</td>
</tr>
<tr>
<td>9</td>
<td>2.047-01</td>
<td>1.950-09</td>
</tr>
<tr>
<td>10</td>
<td>2.046-01</td>
<td>2.096-09</td>
</tr>
<tr>
<td>11</td>
<td>2.045-01</td>
<td>2.416-09</td>
</tr>
<tr>
<td>12</td>
<td>2.043-01</td>
<td>1.193-09</td>
</tr>
<tr>
<td>13</td>
<td>2.042-01</td>
<td>1.673-09</td>
</tr>
<tr>
<td>14</td>
<td>2.041-01</td>
<td>2.139-09</td>
</tr>
<tr>
<td>15</td>
<td>2.039-01</td>
<td>1.892-09</td>
</tr>
<tr>
<td>16</td>
<td>2.038-01</td>
<td>2.765-09</td>
</tr>
<tr>
<td>17</td>
<td>2.037-01</td>
<td>1.455-09</td>
</tr>
<tr>
<td>18</td>
<td>2.036-01</td>
<td>2.474-09</td>
</tr>
<tr>
<td>19</td>
<td>2.034-01</td>
<td>2.387-09</td>
</tr>
<tr>
<td>20</td>
<td>2.033-01</td>
<td>2.241-09</td>
</tr>
<tr>
<td>21</td>
<td>2.032-01</td>
<td>1.426-09</td>
</tr>
<tr>
<td>22</td>
<td>2.031-01</td>
<td>2.012-09</td>
</tr>
<tr>
<td>23</td>
<td>2.029-01</td>
<td>1.892-09</td>
</tr>
<tr>
<td>24</td>
<td>2.028-01</td>
<td>1.688-09</td>
</tr>
<tr>
<td>25</td>
<td>2.027-01</td>
<td>9.022-10</td>
</tr>
<tr>
<td>26</td>
<td>2.025-01</td>
<td>1.973-09</td>
</tr>
<tr>
<td>27</td>
<td>2.024-01</td>
<td>1.313-09</td>
</tr>
<tr>
<td>28</td>
<td>2.023-01</td>
<td>1.639-09</td>
</tr>
<tr>
<td>29</td>
<td>2.022-01</td>
<td>1.513-09</td>
</tr>
<tr>
<td>30</td>
<td>2.021-01</td>
<td>2.561-09</td>
</tr>
</tbody>
</table>

Figure 22. Listing of Time Function at the Output of the AM Demodulator.
Figure 23. Spectrum at the Output of the AM Demodulator.
Figure 24. Spectrum at the Output of the Low Pass Filter.
Figure 25. Block Diagram for FANCAT Analysis of an FM Receiver.
A six-section intermediate frequency Butterworth band-pass filter with a center frequency of 1 MHz and a bandwidth of 80 kHz.

An amplifier with a voltage gain of 140 dB.

A frequency demodulator with a center frequency of 1 MHz.

A ten-section Butterworth low-pass filter with a cutoff frequency of 15 kHz.

The frequency spectrum at the output of each block of the receiver, from block 1 (the output of the signal generator) to block 6 (the output of the frequency demodulator), is shown in Figures 26 through 31. Figure 26 shows the output from the signal generator for a modulating frequency of 10 kHz and a peak frequency deviation of 30 kHz which gives a modulation index of 3.

The output spectrum of the demodulator (Figure 31) shows distortion products produced by the bandpass filters in the receiver. Figure 32 is a time plot of the output of the frequency demodulator. A print of the frequency function at the frequency demodulator output is shown in Figure 33. This print shows the amplitudes and phases of the distortion products over a greater frequency range than the plot because of the limited 50 dB range of the plot.

Figure 34 shows the baseband spectrum at the output of a 10 section Butterworth low-pass filter with a cutoff frequency of 15 kHz. Figure 35 shows a plot of the time waveform at the output of this filter, and Figure 36 is a print of the frequency function at the filter output. It can be seen from Figure 36 that the ten-section Butterworth filter eliminates virtually all distortion products except the 20 kHz component, and it reduces this component's amplitude approximately 60 dB.

The command END OF JOB terminates the program and is followed by the time required to analyze the FM receiver.

D. Conclusions

A new circuit simulation program, FATCAT, for analyzing communications receiver circuits has been developed to an operational state,
Figure 26. Start of FATCAT Run for FM Receiver, Showing Spectrum at the Output of the Signal Generator.
BWNP, 1.87, 80, E3, 1
BWNP, 1.87, 80, E3, 1
PROCESSING COMPLETE THRU BLOCK 2
TPLOTF, 9.9E6, 10.1E6
TPLOTF, 9.9E6, 10.1E6
NSIZE = 21

<table>
<thead>
<tr>
<th>FREQUENCY (MHZ)</th>
<th>DECIBELS</th>
</tr>
</thead>
<tbody>
<tr>
<td>-180</td>
<td>-170</td>
</tr>
<tr>
<td>9.900 I</td>
<td>-</td>
</tr>
<tr>
<td>9.910 I</td>
<td>-</td>
</tr>
<tr>
<td>9.920 I</td>
<td>-</td>
</tr>
<tr>
<td>9.930 I</td>
<td>-</td>
</tr>
<tr>
<td>9.940 I</td>
<td>-</td>
</tr>
<tr>
<td>9.950 I</td>
<td>-</td>
</tr>
<tr>
<td>9.960 I</td>
<td>-</td>
</tr>
<tr>
<td>9.970 I</td>
<td>-</td>
</tr>
<tr>
<td>9.980 I</td>
<td>-</td>
</tr>
<tr>
<td>9.990 I</td>
<td>-</td>
</tr>
<tr>
<td>10.000 I</td>
<td>-</td>
</tr>
<tr>
<td>10.010 I</td>
<td>-</td>
</tr>
<tr>
<td>10.020 I</td>
<td>-</td>
</tr>
<tr>
<td>10.030 I</td>
<td>-</td>
</tr>
<tr>
<td>10.040 I</td>
<td>-</td>
</tr>
<tr>
<td>10.050 I</td>
<td>-</td>
</tr>
<tr>
<td>10.060 I</td>
<td>-</td>
</tr>
<tr>
<td>10.070 I</td>
<td>-</td>
</tr>
<tr>
<td>10.080 I</td>
<td>-</td>
</tr>
<tr>
<td>10.090 I</td>
<td>-</td>
</tr>
<tr>
<td>10.100 I</td>
<td>-</td>
</tr>
<tr>
<td>FREQUENCY</td>
<td></td>
</tr>
</tbody>
</table>

Figure 27. Spectrum at the Output of the First Bandpass Filter.
Figure 28. Spectrum at the Output of the Ideal Multiplier.
Figure 29. Spectrum at the Output of the Second Bandpass Filter.
Figure 30. Spectrum at the Output of the Amplifier.
Figure 31. Spectrum at the Output of the FM Demodulator.
Figure 32. Time Waveform at the Output of the FM Demodulator.
<table>
<thead>
<tr>
<th>LINE</th>
<th>FREQ</th>
<th>REAL</th>
<th>IMAG</th>
<th>MAG</th>
<th>DB</th>
<th>PHASE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1025</td>
<td>0.0000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>164.10</td>
<td>180.0</td>
</tr>
<tr>
<td>1026</td>
<td>1.0000E+04</td>
<td>-1.384</td>
<td>-0.448</td>
<td>1.454</td>
<td>325.25</td>
<td>-162.1</td>
</tr>
<tr>
<td>1027</td>
<td>2.0000E+04</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-108.66</td>
<td>28.8</td>
</tr>
<tr>
<td>1028</td>
<td>3.0000E+04</td>
<td>0.000</td>
<td>0.020</td>
<td>0.020</td>
<td>-33.92</td>
<td>-90.0</td>
</tr>
<tr>
<td>1029</td>
<td>4.0000E+04</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-110.85</td>
<td>-40.1</td>
</tr>
<tr>
<td>1030</td>
<td>5.0000E+04</td>
<td>-0.001</td>
<td>-0.014</td>
<td>0.014</td>
<td>-37.08</td>
<td>-92.3</td>
</tr>
<tr>
<td>1031</td>
<td>6.0000E+04</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-117.33</td>
<td>130.8</td>
</tr>
<tr>
<td>1032</td>
<td>7.0000E+04</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-43.44</td>
<td>166.6</td>
</tr>
<tr>
<td>1033</td>
<td>8.0000E+04</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-124.88</td>
<td>-17.6</td>
</tr>
<tr>
<td>1034</td>
<td>9.0000E+04</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-62.81</td>
<td>82.4</td>
</tr>
<tr>
<td>1035</td>
<td>1.0000E+05</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-113.68</td>
<td>-122.5</td>
</tr>
<tr>
<td>1036</td>
<td>1.1000E+05</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-72.29</td>
<td>-132.0</td>
</tr>
<tr>
<td>1037</td>
<td>1.2000E+05</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-150.36</td>
<td>107.3</td>
</tr>
<tr>
<td>1038</td>
<td>1.3000E+05</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-154.60</td>
<td>-40.4</td>
</tr>
<tr>
<td>1039</td>
<td>1.4000E+05</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-91.80</td>
<td>53.2</td>
</tr>
<tr>
<td>1040</td>
<td>1.5000E+05</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-164.34</td>
<td>-150.7</td>
</tr>
<tr>
<td>1041</td>
<td>1.6000E+05</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-107.43</td>
<td>140.9</td>
</tr>
<tr>
<td>1042</td>
<td>1.7000E+05</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-174.48</td>
<td>47.2</td>
</tr>
<tr>
<td>1043</td>
<td>1.8000E+05</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-110.42</td>
<td>104.0</td>
</tr>
<tr>
<td>1044</td>
<td>1.9000E+05</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>-192.12</td>
<td>100.0</td>
</tr>
</tbody>
</table>

Figure 33. Listing of the Frequency Function at the Output of the FM Demodulator.
BWLOWP, 15*E3, 10.
BWLOWP, 15*E3, 10.
BLOCK, 7
BLOCK, 7
PROCESSING COMPLETE THRU BLOCK 7
TPlotF
TPlotF
ENTER LOW, HIGH FREQUENCIES
0.*.200*E3

NSIZE = 21

<table>
<thead>
<tr>
<th>FREQUENCY (KHZ)</th>
<th>DECIBELS</th>
</tr>
</thead>
<tbody>
<tr>
<td>-40</td>
<td></td>
</tr>
<tr>
<td>-30</td>
<td></td>
</tr>
<tr>
<td>-20</td>
<td></td>
</tr>
<tr>
<td>-10</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td></td>
</tr>
<tr>
<td>120</td>
<td></td>
</tr>
<tr>
<td>130</td>
<td></td>
</tr>
<tr>
<td>140</td>
<td></td>
</tr>
<tr>
<td>150</td>
<td></td>
</tr>
<tr>
<td>160</td>
<td></td>
</tr>
<tr>
<td>170</td>
<td></td>
</tr>
<tr>
<td>180</td>
<td></td>
</tr>
<tr>
<td>190</td>
<td></td>
</tr>
<tr>
<td>200</td>
<td></td>
</tr>
</tbody>
</table>

Figure 34. Spectrum at the Output of the Low Pass Filter.
Figure 35. Time Waveform at the Output of the Low Pass Filter.
Listing of the Frequency Function at the Output of the Low Pass Filter.
although there are many ways in which the system can be improved. The simulation consists of block models of signal sources, filters, amplifiers, mixers, demodulators, etc., along with an operating framework. The program has been fully implemented on both Univac-1108 and SIGMA-5 computers. Use of the present form of the program has shown it to be a useful tool for rapid analysis of communications circuits.

The time and effort available for this work did not permit investigation of numerous ideas for improvement. Additional development work could produce a far more powerful and flexible program. Some of the areas in which additional work should be undertaken are:

(1) The present signal generator block permits simulation of only simple modulation frequencies. Development of a modulator block in which an arbitrary modulating signal could be stored in a data array is needed.

(2) Models for coupled circuits are needed.

(3) Present form of the FM and PM demodulators operate satisfactorily only when the demodulator frequency is perfectly aligned with the carrier frequency, hence no d-c level is generated. Better demodulation models would be desirable.

(4) Control software could be modified to permit insertion, deletion, or replacement of a given block.

(5) A restart capability without the necessity of re-entering the circuit blocks would be helpful.

(6) A routine to print out the circuit description is needed.
IV. TIME DOMAIN SIMULATION

A. Introduction

Digital simulation of a system governed by differential equations can be based upon either a time domain or a frequency domain description of the system. For time domain simulation, the describing differential equations are usually expressed in state variable form and the system response is obtained by numerical integration of the state equations. The time domain simulation method is applicable to both linear and nonlinear systems. For frequency domain simulation, the describing differential equations are expressed in transfer function form, and simulation is accomplished by application of a fast Fourier transform (FFT) algorithm. Frequency domain simulation normally requires less computer time than does time domain simulation, but the transform theory upon which the frequency domain approach is based is limited to linear systems. Accordingly, the simulation of linear systems is usually based upon the frequency domain while that of nonlinear systems is based upon the time domain.

Cascade systems whose only nonlinearities are algebraic are exceptions to the above classification and can be effectively simulated using the FFT algorithm. Since such systems have no feedback, the complete response (for the time range of interest) of the first block or subsystem can be obtained before considering the response of the following blocks. Similarly, the response of the second block can be found after the first block's response has been found. The output of a block represented by an algebraic nonlinearity (e.g., multiplication, saturation, polynomial nonlinearities, etc.) depends only upon the instantaneous input to the block and can be readily determined from the algebraic description of the nonlinearity without recourse to differential equations or frequency domain transfer functions. A cascade system composed of linear dynamic blocks and nonlinear algebraic blocks can be simulated in a serial fashion using the FFT algorithm to transform back and forth between the time and frequency domains as required. This concept has been utilized at the Engineering Experiment Station in the development of the FATCAT simulation described in Section III of this report.
The digital simulation of communication systems with feedback and with nonlinear subsystems requires the use of the complete time domain formulation. The remainder of this section describes the development of a user-oriented communication system block diagram simulation program applicable to cascade or feedback, linear or nonlinear systems. Although the resulting time domain program has general applicability, its use is recommended only for those systems for which the more efficient FFT simulation program (FATCAT) is not applicable.

A number of computer languages have been developed for digital simulation of dynamic systems in recent years. These programming languages, such as MIMIC, DSL-90, CSMP, and CSSL [4], are intended to make it easy for an engineer experienced in analog computer simulation to use a digital computer. The languages have simple input/output instructions and generally have efficient numerical integration algorithms. They can be considered block diagram simulators only in the limiting case where each block is no more complex than a single integration or addition; i.e., only when the block diagram is in reality an analog computer flow diagram. However, at least one of these languages, MIMIC, has a subprogram capability which allows one to construct elementary blocks better suited to communication systems simulation than simple integrators and summers. With these block subprograms the communications system engineer can easily prepare the necessary simulation input data directly from his block diagram. Using the MIMIC simulation language as a basis, a modular time domain simulation program, TIMSIM, has been developed for communication system block diagram simulation. In the following, the basic language MIMIC will be described briefly and its use with TIMSIM will be developed. Application of TIMSIM to the simulation of an AM receiver and an automatic gain control system will also be given.

B. MIMIC: A Continuous System Simulation Language

The simulation language MIMIC was developed at Wright-Patterson AFB in the mid-1960's for the digital simulation of dynamic systems. Detailed instructions for its use are given in the original MIMIC report [5].
and in the previously referenced textbook by Stephenson [4]. Reference should be made to these sources for a more complete description of MIMIC than appears in the following.

The MIMIC language provides a set of functions (including integration) specifically chosen to perform the operations necessary to solve systems of ordinary differential equations. A function is used by listing the name of the output (beginning in column 10) and the name and arguments of the function (beginning in column 19) on standard computer cards. For example, the equation \( x = \log(y) \) would be programmed by punching "X" in column 10 and "LOG(Y)" in columns 19-24. A block-oriented program for simulating a dynamic system is obtained by first drawing a detailed block diagram of the system and then listing the interconnections of the blocks. The block diagram and MIMIC program for a system described by \( \dot{x} + x + x = 0 \) for zero initial conditions is given in Figures 37 and 38. The first four lines in Figure 38 correspond directly to the connections at the four blocks in Figure 37. The name NEG2DX was arbitrarily selected to represent the negative of the second derivative of \( x \). The first box follows from \( -\dot{x} = x + \ddot{x} \), and the second box and instruction reverse the sign of the variable to give \( \ddot{x} \). The third box and instruction correspond to the integration of \( \dot{x} \) to give \( x \) while the fourth ones further integrate \( \dot{x} \) to obtain the variable of interest, \( x \). In the integration instructions, the "0" corresponds to the initial conditions. The detailed block diagram and resulting MIMIC instructions closely follow analog computer programming techniques, but do not require the amplitude scaling and time scaling required for analog computation. The last four MIMIC instructions in Figure 38 are bookkeeping instructions. The FIN(T,10) instruction causes the simulation to stop when time, \( T \), reaches ten seconds. The HDR(TIME,X) instruction establishes the headings on the computer output as TIME and X. The OUT(T,X) instruction means that the variables T and X are to be tabulated as simulation output. The last instruction, END, simply indicates the end of instructions.

The CON and PAR instructions in MIMIC are used to load numerical values for constants and parameters (constants which change from run to run).
Figure 37. Detailed MIMIC Block Diagram for a System Described by $\dot{x} + \ddot{x} + x = 0$.

MIMIC Instructions for Simulation of the System of Figure 37.

10
NEG2DX
2DX
1DX
X
19
ADD(X,1DX)
NEG(NEG2DX)
INT(2DX,0.)
INT(1DX,0.)
FIN(T,10.)
HDR(TIME,X)
OUT(T,X)
END

Figure 38. MIMIC Instructions for Simulation of the System of Figure 37.
run). For example, if the system of Figure 37 had involved a constant, $K_2$, in the differential equation, the instruction "CON($K_2$)", beginning in column 19, would precede the instructions in Figure 38 and the "END" instruction would be followed by a data card with the value of $K_2$ entered in columns 1-12.

As another MIMIC example, consider unity negative feedback around a lowpass filter as shown in Figure 39. The closed loop transfer function is

$$\frac{X}{U} = \frac{K/(s + A)}{1 + K/(s + A)} = \frac{K}{s + (K + A)},$$

and the associated differential equation is

$$\dot{x} = -(K + A) X + Ku.$$  \hspace{1cm} (2)

The corresponding MIMIC program and simulation results for $u(t) = 1$, $K = 1$, and $A = 0.1$ are given in Figure 40.

An alternate method for MIMIC simulation of systems containing frequently occurring subsystems utilizes the subprogram feature of MIMIC. Rather than repeatedly deriving and programming descriptive differential equations for these subsystems, a library of subprograms can be established so that simulation of total systems is essentially reduced to describing the interconnection of the subsystems. Consider the development of a subprogram for the low pass filter previously considered. For subprogram purposes, rename the filter input and output variables as $u_1$ and $X_1$, respectively. The describing differential equation can be shown to be $\dot{X}_1 = K \cdot u_1 - A \cdot X_1$, so that the describing MIMIC equation is

$$X_1 \quad \text{INT}(K \cdot U_1 - A \cdot X_1, 0.).$$  \hspace{1cm} (3)

The associated subprogram is constructed by adding opening and closing instructions which name the subprogram and provide for arbitrary naming of the variables and constants within the subprogram. The resulting subprogram (named FIRST) for the current example is as follows:
Figure 39. Block Diagram of Feedback Example.

---

@GETLIB,MIMIC,IS TEST
MIMIC-03.2-06/09/72-10:32:17

@ADD LARRY,1
1* X INT(-1.1*X+1.0,0.)
2* FIN(T,5.)
3* HDR(TIME,X)
4* OUT(T,X)
5* END

**FURTHER DIAGNOSTICS AND EXECUTION FOLLOW**

<table>
<thead>
<tr>
<th>TIME</th>
<th>X</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>1.00000</td>
<td>9.46962-02</td>
</tr>
<tr>
<td>.20000</td>
<td>.17953</td>
</tr>
<tr>
<td>.30000</td>
<td>.25552</td>
</tr>
<tr>
<td>.40000</td>
<td>.32360</td>
</tr>
<tr>
<td>.50000</td>
<td>.38459</td>
</tr>
<tr>
<td>.60000</td>
<td>.43923</td>
</tr>
<tr>
<td>.70000</td>
<td>.48817</td>
</tr>
<tr>
<td>.80000</td>
<td>.53202</td>
</tr>
<tr>
<td>.90000</td>
<td>.57129</td>
</tr>
<tr>
<td>1.00000</td>
<td>.60648</td>
</tr>
<tr>
<td>1.10000</td>
<td>.63800</td>
</tr>
<tr>
<td>1.20000</td>
<td>.66624</td>
</tr>
<tr>
<td>1.30000</td>
<td>.69154</td>
</tr>
<tr>
<td>1.40000</td>
<td>.71420</td>
</tr>
<tr>
<td>1.50000</td>
<td>.73450</td>
</tr>
<tr>
<td>1.60000</td>
<td>.75269</td>
</tr>
<tr>
<td>1.70000</td>
<td>.76898</td>
</tr>
<tr>
<td>1.80000</td>
<td>.78357</td>
</tr>
<tr>
<td>1.90000</td>
<td>.79665</td>
</tr>
<tr>
<td>2.00000</td>
<td>.80836</td>
</tr>
<tr>
<td>2.10000</td>
<td>.81885</td>
</tr>
<tr>
<td>2.20000</td>
<td>.82825</td>
</tr>
<tr>
<td>2.30000</td>
<td>.83667</td>
</tr>
<tr>
<td>2.40000</td>
<td>.84422</td>
</tr>
<tr>
<td>2.50000</td>
<td>.85097</td>
</tr>
</tbody>
</table>

END 568 MLSEC

---

Figure 40. MIMIC Simulation of System of Figure 39.
The letters BSP and ESP are abbreviations for "begin subprogram" and "end subprogram," respectively, and the arguments of BSP name the input variables and constants while that of ESP names the output variable. In calling such a subprogram from the main MIMIC program, the variable names need not be the same as those used in the subprogram but must be in the same order in the calling statement.

A MIMIC program using the low pass filter subprogram in the simulation of the system of Figure 39 is given in Figure 41. Note that lines 2, 3, and 4 are a listing of the subprogram which has been obtained from a library as a package. Instruction 5 describes the feedback connection for \( u(t) = 1 \), while instructions 6 and 7 are the instructions required to "call" the desired subprogram. The letters CSP and ESP represent "call subprogram" and "end subprogram", respectively. Comparison of the arguments of CSP and BSP shows that the input to the low pass filter is called "E" in the main program and "U1" within the subprogram, and that the constants "K" and "A" are to have the values 1.0 and 0.1, respectively. Note that the name of the subprogram being called is included on the CSP card. Instructions 7, 8, and 9 are not affected by the use of subprograms. Instruction 1 is a parameter statement which must be used to initialize the subprogram output variable for feedback system simulations; the initial value, 0., is entered in columns 1-12 of a data card.

The use of library subprograms to represent subsystems allows the design engineer to digitally simulate his system without a detailed knowledge of programming or of state variable techniques. In the following, a set of library subprograms for communication subsystems will be developed and presented. The composite system is entitled TIMSIM, for time domain simulation.
@ADD LARRY.4
1* FIRST PAR(X)
2* X1 BSP(U1,K,A)
3* FIRST INT(K*U1-A*X1,0.)
4* E ESP(X1)
5* FIRST 1.-X
6* FIRST CSP(E,1.,.1)
7* RSP(X)
8* FIN(T,5.)
9* HDR(TIME,E,:.)
10* OUT(T,E,X)
11* END

**FURTHER DIAGNOSTICS AND EXECUTION FOLLOW***

ENTER DATA NOW

X
0.00000

Figure 41. Subprogram Example.
TIMSIM is not a new digital simulation language to compete with MIMIC, nor is it really a computer program. Rather, it is an operating philosophy and an expandable library of subprograms in the MIMIC language which represent the dynamics of communication subsystems. The communication subsystems modeled for TIMSIM herein by no means exhaust the possibilities but do provide a representative group of subsystems. Included are subprograms which model AM signal generators, bandpass single-tuned filters, bandpass RC filters, mixers, and lowpass RC filters. The mixer subprogram can also be used as a product detector. The method of development of subprograms is described in sufficient detail to allow users to expand the subprogram library to meet their changing needs.

1. Subprogram Derivation

The single-tuned, bandpass filter subprogram STFIL is based upon the transfer function

$$H(s) = \frac{K}{1 + \frac{f_o}{B} \left( \frac{s}{\omega_o} + \frac{\omega_o}{s} \right)} = \frac{X}{U}, \quad (5)$$

as given in Pettit and McWhorter [6], where $B$ is the filter 3 dB bandwidth (Hz), $f_o$ is the filter center frequency (Hz), $K$ is the center frequency gain, and $\omega_o = 2\pi f_o$. The transfer function can be rewritten as

$$\frac{X}{U} = \frac{sK}{s^2 \left( \frac{f_o}{\omega_o} \right) + s + \left( \frac{\omega_o f_o}{B} \right)} \quad (6)$$

from which the corresponding differential equation could be obtained by inspection after cross multiplication with the interpretation of $s^n$ factors as $n^{th}$ derivatives. The resulting differential equation, however, would involve an undesirable derivative of the input, $U$. This undesirable input signal differentiation can be avoided in the same manner as
with analog computer programming [7] by re-expressing the transfer function as

\[ X = \frac{1}{s} \left[ 2\pi B(Ku - X) - \frac{1}{s} (\omega_o^2 X) \right] \]  

(7)

where the complex variables appear only in nested multiplications by 1/s. The desired form of the describing differential equation is obtained by drawing an elementary block diagram for this last expression, selecting the integrator outputs as the state variables, and writing the associated state equations (simultaneous first order differential equations). From Figure 42 the state equations are written as

\[ \dot{x} = 2\pi B(Ku - x) + y \]
\[ \dot{y} = -\omega_o^2 x. \]  

(8)

Figure 42. Single-Tuned Filter Flow Diagram.
Prior to writing the associated MIMIM subprogram, program names must be chosen for the variables. One of the requirements of MIMIC is that variable names not be repeated in other subprograms unless they represent exactly the same quantity. Accordingly, the TIMSIM subprograms are written with variable and constant names which end in a three digit number unique to that subprogram. Further, if a subprogram is repeated in a simulation (e.g., two single-tuned filters in one receiver system), the third digit is increased by one in each additional copy of the subprogram. The variable names selected for the single-tuned filter subprogram STFIL and their relationships to the algebraic variables are as follows: x → OUT110; y → Y110; u → XIN110; K → K110; B → B110; f_o → FRQ110.

2. TIMSIM Subprogram Library.

Table I contains the subprograms used to model AM signal generators (AMSIG, AMSIGS), single-tuned bandpass filters (STFIL), RC bandpass filters (BPFIL), mixers (MIXER), and RC low-pass filters (LPFIL). Included in the table are the definitions of the subprogram input and output variables.

3. Applications

The operation of TIMSIM is best described by examples. The block diagram of Figure 43 describes a hypothetical communication system consisting of an ideal AM signal generator, single-tuned rf filter, mixer (frequency converter), single-tuned i-f filter, product detector, and low pass filter. The corresponding TIMSIM simulation program is given in Figure 44. The first five instructions establish the names and sequence of appearance of constants whose numerical values are to be entered on the data cards. Instructions 6 through 38 present the library subprograms which model the communication subsystem. Note that both the single-tuned filter and the mixer subsystems appear twice in the AM system diagram so that the corresponding subprograms must also appear in duplicate. As mentioned earlier, the repeated use of a
### TABLE I

**TMSIM SUBPROGRAM LIBRARY**

<table>
<thead>
<tr>
<th>Subprogram Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMSIG</td>
<td>Subprogram for Amplitude Modulated Signal Generator. (<strong>BEGIN</strong> subprogram)</td>
</tr>
<tr>
<td>AMSIG5</td>
<td>Subprogram for Sin Amplitude Modulated Signal Generator. (<strong>BEGIN</strong> subprogram)</td>
</tr>
<tr>
<td>STFIL</td>
<td>Subprogram for Single-Tuned Filter. (<strong>BEGIN</strong> subprogram)</td>
</tr>
</tbody>
</table>

**BEGIN SUBPROGRAM FOR AMPLITUDE MODULATED SIGNAL GENERATOR.**

AMSIG BS P(XIN100, FRQ100, M100, K100)

THT.00 6.2831*FRQ100*T

OUT100 K100*(1.+M100*XIN100)*COS(THT100)

AMSIG ESP(OUT100)

**END OF SUBPROGRAM**

**Definitions**

- **XIN100**: Modulating Voltage
- **FRQ100**: Carrier Frequency
- **M100**: Modulation Index
- **K100**: Gain

**BEGIN SUBPROGRAM FOR SIN AMPLITUDE MODULATED SIGNAL GENERATOR.**

AMSIG5 BS P(XIN105, FRQ105, M105, K105)

THT105 6.2831*FRQ105*T

OUT105 K105*(1.+M105*XIN105)*SIN(THT105)

AMSIG5 ESP(OUT105)

**END OF SUBPROGRAM**

**Definitions**

Variables defined similarly to AMSIG above.

**BEGIN SUBPROGRAM FOR SINGLE-TUNED FILTER.**

STFIL BS P(XIN110, B110, FRQ110, K110)

OUT110 INT(Y110+(6.2831*B110)*(K110*XIN110-OUT110),0.)

Y110 INT(-(6.2831*FRQ110)*(6.2831*FRQ110)*OUT110,0.)

STFIL ESP(OUT110)

**END OF SUBPROGRAM**

**Definitions**

- **XIN110**: Input Voltage
- **B110**: Filter Bandwidth, Hz
- **FRQ110**: Filter Center Frequency, Hz
- **K110**: Center Frequency Gain
- **OUT110**: Output Voltage

(Continued)
## TABLE I (Continued)

<table>
<thead>
<tr>
<th>Description</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIXER BSP</td>
<td>OUTI20 = XINI20 * KI20 * SIN(6.2831 * FRQI20 * T)</td>
</tr>
<tr>
<td>MIXER ESP</td>
<td>OUTI20 = KI20 * XINI20 * SIN(6.2831 * FRQI20 * T)</td>
</tr>
<tr>
<td>LPFIL BSP</td>
<td>OUTI30 = INT(6.2831 * FCN130 * (XINI30 - OUTI30), 0.)</td>
</tr>
<tr>
<td>LPFIL ESP</td>
<td>OUTI30 = KI30 * FCN130 * XINI30 + OUTI30</td>
</tr>
<tr>
<td>BPFIL BSP</td>
<td>OUTI40 = INT((K140 * WH140 * XINI40) - (WL140 + WH140) * OUTI40 - Y140), 0)</td>
</tr>
<tr>
<td>BPFIL ESP</td>
<td>OUTI40 = KI40 * WH140 * XINI40 + (WL140 + WH140) * OUTI40 - Y140, 0)</td>
</tr>
</tbody>
</table>

### Definitions

- **XINI20**: Input Voltage
- **FRQI20**: Mixer Product Frequency
- **KI20**: Gain
- **OUTI20**: Output Voltage
- **XINI30**: Input Voltage
- **FCN130**: Corner Frequency, Hz
- **KI30**: DC Gain
- **OUTI30**: Output Voltage
- **XINI40**: Input Voltage
- **FL140**: Lower Corner Frequency, Hz
- **FH140**: Higher Corner Frequency, Hz
- **K140**: Center Frequency Gain
- **OUTI40**: Output Voltage
Figure 43. Hypothetical AM Communication System.
SUBPROGRAM AMSIG

CON (FM, FC, M, K, FO, B)
CON (FMIX, KMIX)
CON (FO1, BO1, KOL)
CON (FMIX1, KMIX1)
CON (FLP, KLP)

BEGIN SUBPROGRAM FOR AMPLITUDE MODULATED SIGNAL GENERATOR.***

AMSIG  BSP(XIN100, FRQ100, M100, K100)
THT100  6.2831*FRQ100*T
OUT100  K100*(1.4*M100*XIN100)*COS(THT100)
AMSIG  ESP(OUT100)

END OF SUBPROGRAM AMSIG********

SUBPROGRAM FOR SINGLE-TUNED FILTER*********

STFIL  BSP(XINIIO, BII0, FRQII0, KII0)
OUTII0  INT(YII0+(6.2831*BII0)*(KII0*XINIIO-OUTII0),0.)
YII0    INT(-(6.2831*FRQII0)*(6.2831*FRQII0)*OUTII0,0.)
STFIL  ESP(OUTII0)

END OF SUBPROGRAM STFIL*********

SUBPROGRAM FOR MIXER*************

MIXER  BSP(XINI20, FRQI20, KI20)
OUTI20  XINI20*KI20*SIN(6.2831*FRQI20*T)
MIXER1  ESP(OUTI20)

END OF SUBPROGRAM MIXER********************

SUBPROGRAM FOR LOW-PASS FILTER**********

LPFIL  BSP(XIN130, FCN130, K130)
OUT130  INT(6.2831*FCN130*(XIN130-OUT130),0.)
LPFIL  ESP(OUT130)

END OF SUBPROGRAM LPFIL**************

SUBPROGRAM FOR SINGLE-TUNED FILTER********

STFILI  BSP(XINIII, BIII, FRQIII, KIII)
OUTIII  INT(YIII+(6.2831*BIII)*(KIII*XINIII-OUTIIi),0.)
YIII    INT(-(6.2831*FRQIII)*(6.2831*FRQIII)*OUTIII,0.)
STFILI  ESP(OUTIII)

END OF SUBPROGRAM STFILI**********

SUBPROGRAM FOR MIXER1*****************

MIXER1  BSP(XINI21, FRQI21, KI21)
OUTI21  XINI21*KI21*SIN(6.2831*FRQI21*T)
MIXER1  ESP(OUTI21)

END OF SUBPROGRAM MIXER1*****************

SUBPROGRAM FOR LOW-PASS FILTER**********

LPFIL  BSP(XIN130, FCN130, K130)
OUT130  INT(6.2831*FCN130*(XIN130-OUT130),0.)
LPFIL  ESP(OUT130)

END OF SUBPROGRAM LPFIL**************

SUBPROGRAM FOR SINGLE-TUNED FILTER********

STFILI  BSP(XINIII, BIII, FRQIII, KIII)
OUTIII  INT(YIII+(6.2831*BIII)*(KIII*XINIII-OUTIIi),0.)
YIII    INT(-(6.2831*FRQIII)*(6.2831*FRQIII)*OUTIII,0.)
STFILI  ESP(OUTIII)

END OF SUBPROGRAM STFILI**********

SUBPROGRAM FOR MIXER1*****************

MIXER1  BSP(XINI21, FRQI21, KI21)
OUTI21  XINI21*KI21*SIN(6.2831*FRQI21*T)
MIXER1  ESP(OUTI21)

END OF SUBPROGRAM MIXER1*****************

SUBPROGRAM FOR LOW-PASS FILTER**********

LPFIL  BSP(XIN130, FCN130, K130)
OUT130  INT(6.2831*FCN130*(XIN130-OUT130),0.)
LPFIL  ESP(OUT130)

END OF SUBPROGRAM LPFIL**************

SUBPROGRAM FOR SINGLE-TUNED FILTER********

STFILI  BSP(XINIII, BIII, FRQIII, KIII)
OUTIII  INT(YIII+(6.2831*BIII)*(KIII*XINIII-OUTIIi),0.)
YIII    INT(-(6.2831*FRQIII)*(6.2831*FRQIII)*OUTIII,0.)
STFILI  ESP(OUTIII)

END OF SUBPROGRAM STFILI**********

SUBPROGRAM FOR MIXER1*****************

MIXER1  BSP(XINI21, FRQI21, KI21)
OUTI21  XINI21*KI21*SIN(6.2831*FRQI21*T)
MIXER1  ESP(OUTI21)

END OF SUBPROGRAM MIXER1*****************

SUBPROGRAM FOR LOW-PASS FILTER**********

LPFIL  BSP(XIN130, FCN130, K130)
OUT130  INT(6.2831*FCN130*(XIN130-OUT130),0.)
LPFIL  ESP(OUT130)

END OF SUBPROGRAM LPFIL**************

SUBPROGRAM FOR SINGLE-TUNED FILTER********

STFILI  BSP(XINIII, BIII, FRQIII, KIII)
OUTIII  INT(YIII+(6.2831*BIII)*(KIII*XINIII-OUTIIi),0.)
YIII    INT(-(6.2831*FRQIII)*(6.2831*FRQIII)*OUTIII,0.)
STFILI  ESP(OUTIII)

END OF SUBPROGRAM STFILI**********

SUBPROGRAM FOR MIXER1*****************

MIXER1  BSP(XINI21, FRQI21, KI21)
OUTI21  XINI21*KI21*SIN(6.2831*FRQI21*T)
MIXER1  ESP(OUTI21)

END OF SUBPROGRAM MIXER1*****************

SUBPROGRAM FOR LOW-PASS FILTER**********

LPFIL  BSP(XIN130, FCN130, K130)
OUT130  INT(6.2831*FCN130*(XIN130-OUT130),0.)
LPFIL  ESP(OUT130)

END OF SUBPROGRAM LPFIL**************

SUBPROGRAM FOR SINGLE-TUNED FILTER********

STFILI  BSP(XINIII, BIII, FRQIII, KIII)
OUTIII  INT(YIII+(6.2831*BIII)*(KIII*XINIII-OUTIIi),0.)
YIII    INT(-(6.2831*FRQIII)*(6.2831*FRQIII)*OUTIII,0.)
STFILI  ESP(OUTIII)

END OF SUBPROGRAM STFILI**********

SUBPROGRAM FOR MIXER1*****************

MIXER1  BSP(XINI21, FRQI21, KI21)
OUTI21  XINI21*KI21*SIN(6.2831*FRQI21*T)
MIXER1  ESP(OUTI21)

END OF SUBPROGRAM MIXER1*****************

SUBPROGRAM FOR LOW-PASS FILTER**********

LPFIL  BSP(XIN130, FCN130, K130)
OUT130  INT(6.2831*FCN130*(XIN130-OUT130),0.)
LPFIL  ESP(OUT130)

END OF SUBPROGRAM LPFIL**************
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>52*</td>
<td>DT1 1./(50.*FM)</td>
</tr>
<tr>
<td>53*</td>
<td>DT2 1./(20.*FC)</td>
</tr>
<tr>
<td>54*</td>
<td>T1 (20./(4.*FM))-DT2</td>
</tr>
<tr>
<td>55*</td>
<td>TF 2./FM</td>
</tr>
<tr>
<td>56*</td>
<td>TTEST T-T1</td>
</tr>
<tr>
<td>57*</td>
<td>TLOGIC FSW(TTEST,TRUE,FALSE,FALSE)</td>
</tr>
<tr>
<td>58*</td>
<td>NTLOG COM(TLOGIC)</td>
</tr>
<tr>
<td>59*</td>
<td>TLOGIC DT EQL(DT1)</td>
</tr>
<tr>
<td>60*</td>
<td>NTLOG DT EQL(DT2)</td>
</tr>
<tr>
<td>61*</td>
<td>END</td>
</tr>
<tr>
<td>62*</td>
<td>HDR(TIME,X,V)</td>
</tr>
<tr>
<td>63*</td>
<td>OUT(T,X,V)</td>
</tr>
</tbody>
</table>

**FURTHER DIAGNOSTICS AND EXECUTION FOLLOW***

ENTER DATA NOW

<table>
<thead>
<tr>
<th>FM</th>
<th>FC</th>
<th>M</th>
<th>K</th>
<th>FO</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000.</td>
<td>1.00000e+06</td>
<td>.50000</td>
<td>1.00000e-03</td>
<td>1.00000e+06</td>
<td>20000.</td>
</tr>
</tbody>
</table>

ENTER DATA NOW

<table>
<thead>
<tr>
<th>FMIX</th>
<th>KMIX</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.45000e+05</td>
<td>10.000</td>
</tr>
</tbody>
</table>

ENTER DATA NOW

<table>
<thead>
<tr>
<th>FO1</th>
<th>BO1</th>
<th>KO1</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.55000e+05</td>
<td>20000.</td>
<td>10.000</td>
</tr>
</tbody>
</table>

ENTER DATA NOW

<table>
<thead>
<tr>
<th>FMIX1</th>
<th>KMIX1</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.55000e+05</td>
<td>100.00</td>
</tr>
</tbody>
</table>

ENTER DATA NOW

<table>
<thead>
<tr>
<th>FLP</th>
<th>KLF</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000.</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

Figure 44. End.
subprogram within a TIMSIM simulation requires renumbering the variable names within each repetition. Accordingly, the second single-tuned filter subprogram is named STFILI and the internal variable names terminate in III rather than in I10. Instructions 39 through 51 constitute the heart of the TIMSIM input; these are the instructions which describe the system block diagram and specify the subsystem parameters through their calling of the subprograms. For instance, instructions 42 and 43 specify that the subsystem is a single-tuned filter whose input is named $y$ (the output of the AM signal generator) and that the filter bandwidth is the input quantity $B$, the center frequency is the input quantity $F_0$, and the center frequency gain is unity. Instruction 43 states that the output of the filter is designated $Z$. Instructions 52-60 result in two separate time spacings between lines of printout; spacing $DT_1$ is used for $T < T_1$ while $DT_2$ is used thereafter. This feature is used only when the simulation results of interest are preceded by a transient of no interest. Instructions 61-64 are as in the MIMIC example described earlier. Figures 45 and 46 present the simulation results in tabular and graphical form.

The block diagram of Figure 47 describes a hypothetical Automatic Gain Control (AGC) system used to demonstrate the application of TIMSIM to feedback systems. The corresponding TIMSIM simulation program is given in Figure 48. The first three instructions describe the constants of the simulation; repeated simulations for new values of the constants in the PAR statements can be made without repeating the program listing. Following the library subprograms (instructions 4-28), the interconnection of the AGC system blocks is described by a series of subprogram call statements. Within this group, instruction 35 gives the effect of the feedback signal upon the forward gain, and instruction 44 demonstrates the use of the internal MIMIC function for a limiter. The remainder of the simulation is analogous to that of the AM system discussed earlier. The simulation results are tabulated in Figure 49.
<table>
<thead>
<tr>
<th>TIME</th>
<th>X</th>
<th>V</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>2.00000-06</td>
<td>.12533</td>
<td>-9.69228-04</td>
</tr>
<tr>
<td>4.00000-06</td>
<td>.24869</td>
<td>-6.16795-03</td>
</tr>
<tr>
<td>6.00000-06</td>
<td>.36812</td>
<td>-1.71452-02</td>
</tr>
<tr>
<td>8.00000-06</td>
<td>.48175</td>
<td>-3.61954-02</td>
</tr>
<tr>
<td>1.00000-05</td>
<td>.58778</td>
<td>-6.83026-02</td>
</tr>
<tr>
<td>1.20000-05</td>
<td>.68454</td>
<td>-.11594</td>
</tr>
<tr>
<td>1.40000-05</td>
<td>.77051</td>
<td>-.17394</td>
</tr>
<tr>
<td>1.60000-05</td>
<td>.84432</td>
<td>-.23469</td>
</tr>
<tr>
<td>1.80000-05</td>
<td>.90482</td>
<td>-.29958</td>
</tr>
<tr>
<td>2.00000-05</td>
<td>.95105</td>
<td>-.38015</td>
</tr>
<tr>
<td>2.20000-05</td>
<td>.98228</td>
<td>-.48357</td>
</tr>
<tr>
<td>2.40000-05</td>
<td>.99803</td>
<td>-.59961</td>
</tr>
<tr>
<td>2.60000-05</td>
<td>.99803</td>
<td>-.70867</td>
</tr>
<tr>
<td>2.80000-05</td>
<td>.98229</td>
<td>-.80481</td>
</tr>
<tr>
<td>3.00000-05</td>
<td>.95106</td>
<td>-.90508</td>
</tr>
<tr>
<td>3.20000-05</td>
<td>.90484</td>
<td>-1.0299</td>
</tr>
<tr>
<td>3.40000-05</td>
<td>.84434</td>
<td>-1.1756</td>
</tr>
<tr>
<td>3.60000-05</td>
<td>.77053</td>
<td>-1.3140</td>
</tr>
<tr>
<td>3.80000-05</td>
<td>.68457</td>
<td>-1.4232</td>
</tr>
<tr>
<td>4.00000-05</td>
<td>.58781</td>
<td>-1.5143</td>
</tr>
<tr>
<td>4.20000-05</td>
<td>.48179</td>
<td>-1.6194</td>
</tr>
<tr>
<td>4.40000-05</td>
<td>.36816</td>
<td>-1.7525</td>
</tr>
<tr>
<td>4.60000-05</td>
<td>.24873</td>
<td>-1.8900</td>
</tr>
<tr>
<td>4.80000-05</td>
<td>.12537</td>
<td>-1.9960</td>
</tr>
<tr>
<td>5.00000-05</td>
<td>4.29471-05</td>
<td>-2.0634</td>
</tr>
<tr>
<td>5.20000-05</td>
<td>-.12529</td>
<td>-2.1229</td>
</tr>
<tr>
<td>5.40000-05</td>
<td>-.24864</td>
<td>-2.2064</td>
</tr>
<tr>
<td>5.60000-05</td>
<td>-.36808</td>
<td>-2.3090</td>
</tr>
<tr>
<td>5.80000-05</td>
<td>-.48171</td>
<td>-2.3936</td>
</tr>
<tr>
<td>6.00000-05</td>
<td>-.58774</td>
<td>-2.4340</td>
</tr>
<tr>
<td>6.20000-05</td>
<td>-.68451</td>
<td>-2.4449</td>
</tr>
<tr>
<td>6.40000-05</td>
<td>-.77048</td>
<td>-2.4639</td>
</tr>
<tr>
<td>6.60000-05</td>
<td>-.84430</td>
<td>-2.5082</td>
</tr>
<tr>
<td>6.80000-05</td>
<td>-.90480</td>
<td>-2.5549</td>
</tr>
<tr>
<td>7.00000-05</td>
<td>-.95104</td>
<td>-2.5694</td>
</tr>
<tr>
<td>7.20000-05</td>
<td>-.98228</td>
<td>-2.5453</td>
</tr>
<tr>
<td>7.40000-05</td>
<td>-.99802</td>
<td>-2.5109</td>
</tr>
<tr>
<td>7.60000-05</td>
<td>-.99803</td>
<td>-2.4963</td>
</tr>
<tr>
<td>7.80000-05</td>
<td>-.98230</td>
<td>-2.4999</td>
</tr>
<tr>
<td>8.00000-05</td>
<td>-.95108</td>
<td>-2.4928</td>
</tr>
<tr>
<td>8.20000-05</td>
<td>-.90486</td>
<td>-2.4537</td>
</tr>
<tr>
<td>8.40000-05</td>
<td>-.84437</td>
<td>-2.3932</td>
</tr>
<tr>
<td>8.60000-05</td>
<td>-.77056</td>
<td>-2.3409</td>
</tr>
<tr>
<td>8.80000-05</td>
<td>-.68460</td>
<td>-2.3125</td>
</tr>
<tr>
<td>9.00000-05</td>
<td>-.58785</td>
<td>-2.2943</td>
</tr>
<tr>
<td>9.20000-05</td>
<td>-.48182</td>
<td>-2.2608</td>
</tr>
<tr>
<td>9.40000-05</td>
<td>-.36820</td>
<td>-2.2047</td>
</tr>
<tr>
<td>9.60000-05</td>
<td>-.24877</td>
<td>-2.1453</td>
</tr>
<tr>
<td>9.80000-05</td>
<td>-.12542</td>
<td>-2.1069</td>
</tr>
<tr>
<td>1.00000-04</td>
<td>-8.59539-05</td>
<td>-2.0929</td>
</tr>
</tbody>
</table>

Figure 45. TIMSIM Results, Hypothetical AM Communication System. (Continued)
<table>
<thead>
<tr>
<th>Value</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.02000-04</td>
<td>.12525</td>
</tr>
<tr>
<td>1.04000-04</td>
<td>.24860</td>
</tr>
<tr>
<td>1.06000-04</td>
<td>.36804</td>
</tr>
<tr>
<td>1.08000-04</td>
<td>.48167</td>
</tr>
<tr>
<td>1.10000-04</td>
<td>.58771</td>
</tr>
<tr>
<td>1.12000-04</td>
<td>.68448</td>
</tr>
<tr>
<td>1.14000-04</td>
<td>.77045</td>
</tr>
<tr>
<td>1.16000-04</td>
<td>.84427</td>
</tr>
<tr>
<td>1.18000-04</td>
<td>.90478</td>
</tr>
<tr>
<td>1.20000-04</td>
<td>.95102</td>
</tr>
<tr>
<td>1.22000-04</td>
<td>.98227</td>
</tr>
<tr>
<td>1.24000-04</td>
<td>.99802</td>
</tr>
<tr>
<td>1.26000-04</td>
<td>.99803</td>
</tr>
<tr>
<td>1.28000-04</td>
<td>.98231</td>
</tr>
<tr>
<td>1.30000-04</td>
<td>.95109</td>
</tr>
<tr>
<td>1.32000-04</td>
<td>.90488</td>
</tr>
<tr>
<td>1.34000-04</td>
<td>.84439</td>
</tr>
<tr>
<td>1.36000-04</td>
<td>.77059</td>
</tr>
<tr>
<td>1.38000-04</td>
<td>.68463</td>
</tr>
<tr>
<td>1.40000-04</td>
<td>.58788</td>
</tr>
<tr>
<td>1.42000-04</td>
<td>.48186</td>
</tr>
<tr>
<td>1.44000-04</td>
<td>.36824</td>
</tr>
<tr>
<td>1.46000-04</td>
<td>.24881</td>
</tr>
<tr>
<td>1.48000-04</td>
<td>.12546</td>
</tr>
<tr>
<td>1.50000-04</td>
<td>1.29795-04</td>
</tr>
<tr>
<td>1.52000-04</td>
<td>-.12520</td>
</tr>
<tr>
<td>1.54000-04</td>
<td>-.24860</td>
</tr>
<tr>
<td>1.56000-04</td>
<td>-.36800</td>
</tr>
<tr>
<td>1.58000-04</td>
<td>-.48163</td>
</tr>
<tr>
<td>1.60000-04</td>
<td>-.58767</td>
</tr>
<tr>
<td>1.62000-04</td>
<td>-.68444</td>
</tr>
<tr>
<td>1.64000-04</td>
<td>-.77042</td>
</tr>
<tr>
<td>1.66000-04</td>
<td>-.84425</td>
</tr>
<tr>
<td>1.68000-04</td>
<td>-.90476</td>
</tr>
<tr>
<td>1.70000-04</td>
<td>-.95101</td>
</tr>
<tr>
<td>1.72000-04</td>
<td>-.98226</td>
</tr>
<tr>
<td>1.74000-04</td>
<td>-.99802</td>
</tr>
<tr>
<td>1.76000-04</td>
<td>-.99804</td>
</tr>
<tr>
<td>1.78000-04</td>
<td>-.98232</td>
</tr>
<tr>
<td>1.80000-04</td>
<td>-.95110</td>
</tr>
<tr>
<td>1.82000-04</td>
<td>-.90489</td>
</tr>
<tr>
<td>1.84000-04</td>
<td>-.84441</td>
</tr>
<tr>
<td>1.86000-04</td>
<td>-.77062</td>
</tr>
<tr>
<td>1.88000-04</td>
<td>-.68467</td>
</tr>
<tr>
<td>1.90000-04</td>
<td>-.58792</td>
</tr>
<tr>
<td>1.92000-04</td>
<td>-.48190</td>
</tr>
<tr>
<td>1.94000-04</td>
<td>-.36828</td>
</tr>
<tr>
<td>1.96000-04</td>
<td>-.24886</td>
</tr>
<tr>
<td>1.98000-04</td>
<td>-.12550</td>
</tr>
<tr>
<td>2.00000-04</td>
<td>-.174173-04</td>
</tr>
<tr>
<td>2.02000-04</td>
<td>.12516</td>
</tr>
</tbody>
</table>
Figure 47. Hypothetical Automatic Gain Control System.
CON(FC, KP, BW, K1, VSET)
PAR(X4, FCR, KP, TPOINT, TF, KFORWD)
PAR(FM)

BEGIN SUBPROGRAM FOR LOW-PASS FILTER

LPFIL BSP(XIN130, FCN130, K130)
OUT130 INT(6.2831*FCN130*(XIN130-OUT130), 0.)
LPFIL ESP(OUT130)

**END OF SUBPROGRAM**

BEGIN SUBPROGRAM FOR SINGLE-TUNED FILTER

STFIL BSP(XIN110, BI10, FRQ110, KI10)
OUT110 INT(Y110+(6.2831*BI10)*(KI10*XIN110-OUT110), 0.)
Y110 INT(-(6.2831*FRQ110)*(6.2831*FRQ110)*OUT110, 0.)
STFIL ESP(OUT110)

***END OF SUBPROGRAM***

BEGIN SUBPROGRAM FOR RC BANDPASS FILTER

BPFIL BSP(XIN140, FLI140, FH140, KI140)
WLI140 6.2831*FL140
WHI140 6.2831*FH140
Y140 INT(WL140*WH140*OUT140, 0.)
OUT140 INT(K140*WH140*XIN140-(WLI140+WHI140)*OUT140-Y140, 0.)
BPFIL ESP(OUT140)

***END OF SUBPROGRAM***

XIN SIN(6.2831*FM*T)
AMSIG5 BSP(XIN105, FRQ105, M105, KI105)
TH105 6.2831*FRQ105*T
OUT105 K105*(1.4*M105*XIN105)*SIN(TH105)
AMSIG5 ESP(OUT105)

***BEGIN SUBPROGRAM FOR RC BANDPASS FILTER***

BPFIL BSP(XIN140, FLI140, FH140, KI140)
WLI140 6.2831*FL140
WHI140 6.2831*FH140
Y140 INT(WL140*WH140*OUT140, 0.)
OUT140 INT(K140*WH140*XIN140-(WLI140+WHI140)*OUT140-Y140, 0.)
BPFIL ESP(OUT140)

XIN SIN(6.2831*FM*T)
AMSIG5 BSP(XIN, FC, .95, 1.)
STFIL BSP(XIN, BW, FC, 1.)
STFIL ESP(X)
X1 KP*X
KVAR 1. - K1*VF
X2 KVAR*X1*KFORWD
X3 X2*SIN(6.2831*FC*T)
BPFIL BSP(X3, .5*FM, .2*FC, 1.)
RSP(X5)
LPIF LSP(X4)
VF1 KP*(X4-VSET)
VF LIMP .9/K1
VF LIM(VF1, -VFLIM, VFLIM)

Figure 48. TIMSIM Listing, Automatic Gain Control System.
(Continued).
45*  DT1  0.05/FCR
46*  DT2  .50/FC
47*  T1   TPOINT - DT2
48*  TTEST T-T1
49*  TLOGIC FSW(TTEST,TRUE,FALSE,FALSE)
50*  NTLOG COM(TLOGIC)
51*  TLOGIC DT EQL(DT1)
52*  NTLOG DT EQL(DT2)
53*  FIN(T,TF)
54*  HDR(TIME,X3,X4,VF,X5)
55*  OUT(T,X3,X4,VF,X5)
56*  END

**FURTHER DIAGNOSTICS AND EXECUTION FOLLOW***

ENTER DATA NOW

<table>
<thead>
<tr>
<th>FC</th>
<th>KP</th>
<th>BW</th>
<th>K1</th>
<th>VSET</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000</td>
<td>2.00000-03</td>
<td>5000.0</td>
<td>1.00000-01</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

ENTER DATA NOW

<table>
<thead>
<tr>
<th>X4</th>
<th>FCR</th>
<th>KP</th>
<th>TPOINT</th>
<th>TF</th>
<th>KFORWD</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00000</td>
<td>100.00</td>
<td>3.0000</td>
<td>4.00000-03</td>
<td>6.00000-03</td>
<td>10000.0</td>
</tr>
</tbody>
</table>

ENTER DATA NOW

| FM    | 1000.0 |

Figure 48. End.
<table>
<thead>
<tr>
<th>TIME</th>
<th>X3</th>
<th>X4</th>
<th>VF</th>
<th>X5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>-3.0000</td>
<td>-3.42534-02</td>
</tr>
<tr>
<td>5.00000-04</td>
<td>-4.0760-05</td>
<td>2.8968</td>
<td>5.6903</td>
<td>-3.42534-02</td>
</tr>
<tr>
<td>1.00000-03</td>
<td>1.02872-04</td>
<td>2.6581</td>
<td>4.9742</td>
<td>-1.5398</td>
</tr>
<tr>
<td>1.50000-03</td>
<td>-6.78247-05</td>
<td>3.4236</td>
<td>7.2767</td>
<td>-3.8317</td>
</tr>
<tr>
<td>2.00000-03</td>
<td>1.84120-04</td>
<td>2.9041</td>
<td>7.5124</td>
<td>-6.3602</td>
</tr>
<tr>
<td>2.50000-03</td>
<td>-9.51716-05</td>
<td>3.4685</td>
<td>7.4054</td>
<td>-1.71602-02</td>
</tr>
<tr>
<td>3.00000-03</td>
<td>2.84755-04</td>
<td>2.9241</td>
<td>5.7723</td>
<td>-4.7952</td>
</tr>
<tr>
<td>3.50000-03</td>
<td>-1.15230-04</td>
<td>3.4720</td>
<td>7.4159</td>
<td>3.04278-02</td>
</tr>
<tr>
<td>4.00000-03</td>
<td>3.95770-04</td>
<td>2.9258</td>
<td>5.7773</td>
<td>-4.6301</td>
</tr>
<tr>
<td>4.05000-03</td>
<td>3.01825-04</td>
<td>2.9397</td>
<td>5.8191</td>
<td>3.4545</td>
</tr>
<tr>
<td>4.10000-03</td>
<td>1.84063-04</td>
<td>2.9858</td>
<td>5.9573</td>
<td>1.1311</td>
</tr>
<tr>
<td>4.15000-03</td>
<td>6.21273-05</td>
<td>3.0570</td>
<td>6.1709</td>
<td>1.7666</td>
</tr>
<tr>
<td>4.20000-03</td>
<td>-5.57693-05</td>
<td>3.1427</td>
<td>6.4279</td>
<td>2.1556</td>
</tr>
<tr>
<td>4.25000-03</td>
<td>-1.27702-04</td>
<td>3.2317</td>
<td>6.6947</td>
<td>2.2567</td>
</tr>
<tr>
<td>4.30000-03</td>
<td>-7.19128-04</td>
<td>3.3140</td>
<td>6.9417</td>
<td>2.0872</td>
</tr>
<tr>
<td>4.35000-03</td>
<td>-2.00990-04</td>
<td>3.3825</td>
<td>7.1474</td>
<td>1.7065</td>
</tr>
<tr>
<td>4.40000-03</td>
<td>-1.97042-04</td>
<td>3.4329</td>
<td>7.2985</td>
<td>1.1917</td>
</tr>
<tr>
<td>4.45000-03</td>
<td>-1.71074-04</td>
<td>3.4630</td>
<td>7.3889</td>
<td>.61588</td>
</tr>
<tr>
<td>4.50000-03</td>
<td>-1.25771-04</td>
<td>3.4723</td>
<td>7.4168</td>
<td>3.57353-02</td>
</tr>
<tr>
<td>4.55000-03</td>
<td>-6.23912-05</td>
<td>3.4610</td>
<td>7.3829</td>
<td>-5.1181</td>
</tr>
<tr>
<td>4.60000-03</td>
<td>1.81439-05</td>
<td>3.4298</td>
<td>7.2894</td>
<td>-1.0048</td>
</tr>
<tr>
<td>4.65000-03</td>
<td>1.14925-04</td>
<td>3.3801</td>
<td>7.1404</td>
<td>-1.6290</td>
</tr>
<tr>
<td>4.70000-03</td>
<td>2.24564-04</td>
<td>3.3142</td>
<td>6.9426</td>
<td>-1.7688</td>
</tr>
<tr>
<td>4.75000-03</td>
<td>3.39990-04</td>
<td>3.2358</td>
<td>6.7074</td>
<td>-2.0009</td>
</tr>
<tr>
<td>4.80000-03</td>
<td>4.49451-04</td>
<td>3.1508</td>
<td>6.4525</td>
<td>-2.0913</td>
</tr>
<tr>
<td>4.85000-03</td>
<td>5.37047-04</td>
<td>3.0675</td>
<td>6.2026</td>
<td>-2.0006</td>
</tr>
<tr>
<td>4.90000-03</td>
<td>5.85543-04</td>
<td>2.9958</td>
<td>5.9874</td>
<td>-1.6962</td>
</tr>
<tr>
<td>4.95000-03</td>
<td>5.81355-04</td>
<td>2.9459</td>
<td>5.8376</td>
<td>-1.1710</td>
</tr>
<tr>
<td>5.00000-03</td>
<td>5.20045-04</td>
<td>2.9259</td>
<td>5.7778</td>
<td>-4.6135</td>
</tr>
<tr>
<td>5.05000-03</td>
<td>4.09642-04</td>
<td>2.9399</td>
<td>5.8195</td>
<td>.34709</td>
</tr>
<tr>
<td>5.10000-03</td>
<td>2.68857-04</td>
<td>2.9859</td>
<td>5.9577</td>
<td>1.1327</td>
</tr>
<tr>
<td>5.15000-03</td>
<td>1.21864-04</td>
<td>3.0571</td>
<td>6.1712</td>
<td>1.7682</td>
</tr>
<tr>
<td>5.20000-03</td>
<td>-9.49593-06</td>
<td>3.1428</td>
<td>6.4282</td>
<td>2.1572</td>
</tr>
<tr>
<td>5.25000-03</td>
<td>-1.10613-04</td>
<td>3.2317</td>
<td>6.6950</td>
<td>2.2584</td>
</tr>
<tr>
<td>5.30000-03</td>
<td>-1.75682-04</td>
<td>3.3140</td>
<td>6.9419</td>
<td>2.0888</td>
</tr>
<tr>
<td>5.35000-03</td>
<td>-2.05506-04</td>
<td>3.3826</td>
<td>7.1476</td>
<td>1.7080</td>
</tr>
<tr>
<td>5.40000-03</td>
<td>-2.04383-04</td>
<td>3.4329</td>
<td>7.2987</td>
<td>1.1931</td>
</tr>
<tr>
<td>5.45000-03</td>
<td>-1.76745-04</td>
<td>3.4630</td>
<td>7.3891</td>
<td>.61719</td>
</tr>
<tr>
<td>5.50000-03</td>
<td>-1.25781-04</td>
<td>3.4723</td>
<td>7.4170</td>
<td>3.69046-02</td>
</tr>
<tr>
<td>5.55000-03</td>
<td>-5.30914-05</td>
<td>3.4610</td>
<td>7.3830</td>
<td>-5.1079</td>
</tr>
<tr>
<td>5.60000-03</td>
<td>4.04190-05</td>
<td>3.4299</td>
<td>7.2896</td>
<td>-1.0039</td>
</tr>
<tr>
<td>5.65000-03</td>
<td>1.53555-04</td>
<td>3.3802</td>
<td>7.1405</td>
<td>-1.4282</td>
</tr>
<tr>
<td>5.70000-03</td>
<td>2.82489-04</td>
<td>3.3142</td>
<td>6.9427</td>
<td>-1.7683</td>
</tr>
<tr>
<td>5.75000-03</td>
<td>4.19098-04</td>
<td>3.2358</td>
<td>6.7075</td>
<td>-2.0004</td>
</tr>
<tr>
<td>5.80000-03</td>
<td>5.49811-04</td>
<td>3.1509</td>
<td>6.4527</td>
<td>-2.0910</td>
</tr>
<tr>
<td>5.85000-03</td>
<td>6.56145-04</td>
<td>3.0676</td>
<td>6.2027</td>
<td>-2.0004</td>
</tr>
<tr>
<td>5.90000-03</td>
<td>7.17897-04</td>
<td>2.9958</td>
<td>5.9875</td>
<td>-1.6960</td>
</tr>
<tr>
<td>5.95000-03</td>
<td>7.18865-04</td>
<td>2.9459</td>
<td>5.8377</td>
<td>-1.1707</td>
</tr>
<tr>
<td>6.00000-03</td>
<td>6.53252-04</td>
<td>2.9260</td>
<td>5.7779</td>
<td>-4.6095</td>
</tr>
<tr>
<td>6.05000-03</td>
<td>5.29630-04</td>
<td>2.9399</td>
<td>5.8196</td>
<td>.34764</td>
</tr>
</tbody>
</table>

Figure 49. TIMSIM Results, AGC System.
D. Conclusions

As stated earlier, TIMSIM is basically a simulation philosophy rather than a program or language; it has been presented here within the framework of the simulation language MIMIC. Several subprograms representative of communication system building blocks have been presented and the method of generation of subprograms has been demonstrated so that additional subprograms can be generated as desired by the user. It is worth noting that the TIMSIM concept is also applicable to non-communication systems. Appropriate subprograms can be developed for mechanical systems, and control systems, etc.
V. REFERENCES


LISTING OF SOFTWARE MODIFICATION TO CIRCUIT TO WRITE OUTPUT DATA INTO EXTERNAL FILE.
SUBROUTINE LINK6A(PLTX, NPNT, TIME, DUMY2, LFLAG, DUMY3, NMP, 
1  DEI'T, NREAD )
INTEGER HOUR(2), DAY(2)
DIMENSION WORD(1), PLOT(NPNT,1), TIME(1)
DIMENSION WORD(1)
COMMON WORD M1, M5, M6, LPDS, LDS, TITLE(12)
COMMON NDPLT, IFPLT, NPNT, KPRNT, DUMY1, DTCM,
COMMON /SCRCTC/ TITLE(120), AMAX, AINV, DAY, HOUR, IBAK, IMAX,
1  IMIN, KFLASE, NN, J, NWIPE, PLT1, PLT2, PLT3, PLT4,
2  XPLTS, A
EQUIVALENCE ( WORD(126), N7 ) ( WORD(38), N2 )
EQUIVALENCE ( WORD(33), LINTVL )
EQUIVALENCE ( WORD(33), LKPO )
EQUIVALENCE ( WORD(130), LMAX )
EQUIVALENCE ( WORD(31), LMUK )
EQUIVALENCE ( WORD(111), NPCEL )
EQUIVALENCE ( WORD(112), LPLT )
NRF = 51
NG = 21
IF ( LFLAG = 1 ) 390, 385, 390
390 CONTINUE
LFLAG = 2
ARITE (N7) ( PLOT(1,1), I=1, NPNT )
READ = NWIPE
READ = N7
READ (N2) XPLTS, ( TITLE(I), I=1, NPLTS )
390 CONTINUE
SRRITE (N6, 33)
C**** READ TIME AND PLOT VARIABLES FOR NPNT VALUES OF TIME.
NL = NP1
NPL = NPCEL/6
DO 210 JX=1, NPL
NL = NL + 6* (* NPNTS + 1 ) - 1
210 CONTINUE
NL = NPLS + 1
NL = NP1
DO 230 JX=1, NPNT
TIME(JX) = PLOT(NWL, 1)
DO 220 I=1, NPLTS
NL = NWL + 1
PLOT(JX, I) = PLOT(NWL, 1)
220 CONTINUE
NL = NWL + 1
230 CONTINUE
CALL CLOCK (6HRSTIME, IDUM)
DO 210 J=1, NPLTS
REF = PLOT(1, J)
AMAX = REF
AMIN = REF
210 CONTINUE
LINK6A (Continued)

C**** FIND MAXIMUM AND MINIMUM OF EACH VARIABLE
DO 41 I=1, NPNT
    IF (AMAX.GT.PLOT(I,J)) GO TO 40
    AMAX = PLOT(I,J)
    IMAX = I
40 IF (AMIN.LT.PLOT(I,J)) GO TO 41
    AMIN = PLOT(I,J)
    IMIN = I
41 CONTINUE
    IF (ABS(AMAX-REF).GT.ABS(AMIN-REF)) GO TO 42
C**** USE *FINDER* TO DETERMINE 10 - 90 PERCENT RISE AND FALL
C**** TIMES FOR PREDOMINATELY POSITIVE VARIABLES.
    PT1 = .1*(AMAX-REF)+REF
    CALL FINDER ( I_MIN, NPNT, I, IVAX, I_MIN, T1, T2, PLOT(I,J), TIME )
    PT1 = .9*(AMAX-REF)+REF
    CALL FINDER ( I_MIN, NPNT, I, IMIN, I_MIN, T3, T4, PLOT(I,J), TIME )
42 CONTINUE
C**** USE *FINDER* TO DETERMINE 10 - 90 PERCENT RISE AND FALL
C**** TIMES FOR PREDOMINATELY POSITIVE VARIABLES.
    PT1 = .1*(AMAX-REF)+REF
    CALL FINDER ( I, I_MAX, I_MIN, NPNT, T1, T4, PLOT(I,J), TIME )
    PT1 = .9*(AMAX-REF)+REF
    CALL FINDER ( I, I_MAX, I_MIN, NPNT, T2, T3, PLOT(I,J), TIME )
43 CONTINUE
C**** WRITE OUT RISE AND FALL TIMES.
C ** COMMENT BELOW Suppresses PRINT OF RISE AND FALL TIMES.
C WRITE (N6,5C) TITLE(J), T1, T2, T3, T4
C ** *
21 CONTINUE
C**** IF ONLY RESPONSE TIMES DESIRED, RETURN.
    IF (I(PLOT).EQ.0.4) GO TO 80
C C INSERT PLOTTING ROUTINES HERE.
    CALL PLTDTA(PLOT(NVP,1), NR, NC, TIME, PLOT, NPNT, NPLTS, 1 Title, DFLT, NPNT)
C C CALL CLOCK (6H*PLOT*1DUM)
C ** NEXT TWO COMMENTS Suppress LINE PRINTER PLOTS.
C CALL PLOTER ( PLOT(NWP,1), NR, NC, TIME, PLOT, NPNT, NPLTS, 1 TITLE, DFLT, NPNT)
C C 1 NPNT )
C ** *
C CALL CLOCK (6H*PLOT*1IDUM)
80 CONTINUE
RETURN
30 FORMAT(////52X15RESPONSE TIMES///31X...PARAMETER4X
     1 8H10 RISE 4X8H90 RISE4X8H10 FALL•X;H00 FALL/)##
50 FORMAT(33XA6,1X1P4E12.4)
END
SUBROUTINE PLTDATA(NSCR, NR, NC, TIME, PLOT, NPNT, NPLOTS)
DIMENSION PLOT(NDIM, NPLOTS), TITLE(NPLOTS)
DIMENSION MSCS(NR, NC), TIME(NPNT)
WRITE(19) NPNT
WRITE(19) NPLOTS
WRITE(19) TIME
WRITE(19) PLOT
WRITE(19) TITLE
RETURN
END
APPENDIX B

Listing of Software for Displaying Data Extracted from CIRCUS.

NOTE: Many of the Hollerith strings in format statements were delimited with quote marks. The printer used to make the following listing did not have the quote character; a minus sign appears where each quote should have been.
MAIN PROGRAM

PARAMETER N = 32
DIMENSION TIME(5000), PLAT(5000)
COMPLEX A(N)
DIMENSION IBUF(10000)
LOGICAL CPLT, PDPRT, DOMFLG
CPLT = .FALSE.
PDPRT = .FALSE.
DOMFLG = .TRUE.
READ(19) NPNT
WRITE(6,200) NPNT
NP = N
WRITE(6,202) NP
READ(19) NPLOTS
WRITE(6,201) NPLOTS
CALL RDTF(NPNT,NPLOTS,TIME,PLAT)
IGAM = ALOG(N) / ALOG(2.) + .1
1 WRITE(6,101)
WRITE(6,1001)
WRITE(6,231)
READ(5,100,ERR=1,END=999) ISWTCH
IF(ISWTCH .GT. 0) GO TO 2
IF(ISWTCH .LT. 0) GO TO 3
CALL WRTDTA(NPNT,NPLOTS,TIME,PLAT)
GO TO 1
2 IF(ISWTCH .GT. NPLOTS) GO TO 5C
WRITE(6,102)
READ(5,100) ISTART
DO 10 I = 1,N
ITQ = I + ISTART - 1 + (ISWTCH - 1) * NPNT
10 A(1) = CMPLEX(PLAT(ITQ), 0.)
TP = ((TIME(ISTART + N - 1) - TIME(ISTART)) / (N -1)) * N
IF(.NOT. PDPRT) WRITE(6,203) TP
PDPRT = .TRUE.
CALL FFT(A,IGAM,-1)
CALL LFOLD(A,N)
WRITE(6,103)
READ(5,100) JSWTCH
IF(JSWTCH .LT. 100) GO TO 20
CALL PRNT(A,N)
JSWTCH = JSWTCH - 100
MAIN (Continued)

20 IF(JSWTCH *EQ. 0) GO TO 1
   DELF = 1. / TP
   IF(JSWTCH *LT. 10) GO TO 30
   IF(.NOT. CPLT) CALL PLOTS(IBUF(1),10000,2)
   CPLT = .TRUE.
   WRITE(6,108)
   READ(5,100) FMAX
   CALL CPOI(TF(A,N,DELF,FMAX))
   JSWTCH = JSWTCH - 10
   IF(JSWTCH *EQ. 0) GO TO 1
30 CONTINUE
   WRITE(6,104)
   READ(5,100) FLO
   WRITE(6,105)
   READ(5,100) FHI
   CALL TTFP(A,N,DELF,FLO,FHI)
   GO TO 1
3 ISWTCH = ABS(ISWTCH)
   IF(ISWTCH *GT. NPLOTS) GO TO 50
   WRITE(6,102)
   READ(5,100) ISTART
   DO 11 I = 1,N
      ITQ = I + ISTART - 1 + (ISWTCH - 1) * NPNT
11 A(I) = CMPLX(PLAT(ITQ), 0*)
      TP = ((TIME(1) - TIME(N)) / (N - 1)) * N
      IF(.NOT. PDPR) WRITE(6,203) TP
      PDPR = .TRUE.
      WRITE(6,103)
   READ(5,100) JSWTCH
   IF(JSWTCH *LT. 10) GO TO 21
   CALL PRNT(A,N)
   JSWTCH = JSWTCH - 10
21 IF(JSWTCH *EQ. 0) GO TO 1
   IF(JSWTCH *LT. 10) GO TO 31
   IF(.NOT. CPLT) CALL PLOTS(IBUF(1),10000,2)
   CPLT = .TRUE.
C **
C  PLACE CALCOMP TIME PLOTS HERE
   WRITE(6,900)
C **
MAIN (Continued)

JSWITCH = JSWITCH - 10
IF(JSWITCH .EQ. 0) GO TO 1
31 CONTINUE
WRITE(6,304) NP
WRITE(6,3C3)
READ(5,100) NSTART
NST = NSTART
WRITE(6,3C5)
READ(5,100) NSTOP
NSP = NSTOP
WRITE(6,306)
READ(5,1U0) NJUMP
CALL TTPP(A,NST,NSP,NJUMP)
GO TO 1
50 WRITE(6,107) NPLOTS
GO TO 1
100 FORMAT( )
101 FORMAT( - ENTER 0 FOR PRINT OF DATA SET FROM CIRCUS- )
1001 FORMAT( - ENTER PLUS DATA SET NUMBER FOR FREQ FCN- )
2001 FORMAT( - ENTER MINUS DATA SET NUMBER FOR TIME FCN- )
102 FORMAT( - ENTER ISTART- )
103 FORMAT( - ENTER 100 FOR PRINT, 010 FOR CALCOMP PLOT- ,
A = OR 001 FOR TTY PLOT- )
104 FORMAT( - ENTER FLG- )
105 FORMAT( - ENTER FHI- )
107 FORMAT( // - ERROR--LARGEST DATA SET NUMBER IS -12/ )
108 FORMAT( - ENTER THE HIGHEST DESIRED FREQ IN THE SPECTRUM, FMAX-/ )
200 FORMAT( - THE NUMBER OF POINTS PER DATA SET = -15 )
201 FORMAT( - THE NUMBER OF DATA SETS = -12/ )
202 FORMAT( - THE NUMBER OF POINTS USED IN A TRANSFORM = -15 )
203 FORMAT( - THE PERIOD OF THE TIME FUNCTION = -1PE11.4, SEC-/ )
303 FORMAT( - ENTER NSTART- )
304 FORMAT( - ARRAY SIZE = -15, NSTOP MUST BE EQUAL TO-
A = OR LESS THAN THIS VALUE-/ )
305 FORMAT( - ENTER NSTOP- )
306 FORMAT( - ENTER NJUMP- )
900 FORMAT( - CALCOMP TIME PLOT NOT OPERATIONAL-/ )
999 IF(CPLT) CALL PLOT(_,DO,9999)
STOP
END
SUBROUTINE (LOTFE(A,N,DELX,FMAX)

COMMENT - THE FOLLOWING CONTROL STATEMENT MUST PRECEDE THE EXECUTE STATEMENT FOR RUNS USING CALCOMP PLOTS.

COMMON, I, J, I, J, J, J, J, J

USE UNIT #, TF5

COMMON /A(1)/
IR = FAX / DELX + .01
FMAX = IR * DELX
FLC = - FMAX
FHI = FMAX

1790 FORMAT(1)
XTEST = ABS(FH1-FLC)
IF(XTEST.LT.1.E-3) GO TO 9999
XTEST = XTEST/(ABS(FLC) + ABS(FH1))
IF(XTEST.LT.1.E-3) GO TO 9999
YSPRD = 7.
R2 = X/2
NST = N2 - 1R + 1
NSP = N2 + 1R + 1
T1 = 1.E-35
DO 4 I = NST, NSP
T2 = CAS(4(I))
4 ..., IF(T2.GT.T1) T1 = T2
DJMAX = 22. ALOG1 (T1)
CALL SCALE(FMAX, FAXSCL)
XWXSCL = FAXSCL
XXSCAL = 1.**((X马克思 / 20.))
CALL FACTOR(1.4)
CALL PLOT(0...-2...+3)
CALL PLOT(12...+3)
CALL PLOT(-1...-14...+3)
DO 3 I = 1,2
CALL PLOT(-1...+2)
CALL PLOT(1...+2)
CALL PLOT(1...-14...+2)
CALL PLOT(-1...+10...+2)
CALL PLOT(1...+1...+2)
CALL PLOT(1...-14...+2)
CALL PLOT(-1...-14...+2)
DO 3000 IFAGAIN = 1,2

C **
DO LOOP 1 TO CALIBRATES THE LEFT HANDER

C **
DO 1 I = 1,75
Y = -14. + 2.*I
IF (YEQ(15) .NOT. 1 GO TO 6
CALL PLOT(-10.1,Y,3)
GO TO 8
6 IF (MOD(I,10) *EQ. 0) GO TO 7
CALL PLOT(-10.15,Y,3)
GO TO 8
7 CALL PLOT(-1.2,Y,3)
8 CALL PLOT(-1.2,Y,2)
1 CONTINUE
Y = 2.2
DO 15 I = 1,8
   J = I - 1
   Y = Y - 2.
   XX = X*Y/
   15 CALL SYMBOL(-11.24*8421.14*90.0.14)
      FCEMT = (FLO + FHI) / 2.
      FUPPER = FHI - FCEMT
      IF (FUPPER > X) GO TO 210C
      IEXP = ALOG10(FUPPER) - 1
      GO TO 210C
210 CONTINUE
      XIEXP = ALOG10(FUPPER)
      IEXP = XIEXP
      REXP = IEXP
      IF (ASS(XIEXP - REXP) *LT. 1.E-2L) AND (XIEXP *GE.* REXP)
        1 IFEXP = IEXP - 1
2101 CONTINUE
      FULSCL = FUPPER * (1. **(-IEXP))
      IFLSCL = FULSCL
      ITEMP = 1. * FULSCL
      RITEMP = ITEMP
      SCAL1 = IFLSCL
      SCAL1 = 1. * SCALE1 / FULSCL
      ITEMP = 10. * IFLSCL
      SCALE1 = SCALE1 / ITEMP
C ** DO LOOP 40 CALIBRATES THE BOTTOM POSITIVE BORDER
C **
   DO 40 I = 1,ITEMP
      X = I * SCALE1
      IF (MOD(I,5) *EQ. 0) GO TO 32
      CALL PLOT(X, -14.1*3)
      GO TO 36
32 IF (MOD(I,10) *EQ. 0) GO TO 34
      CALL PLOT(X, -14.16*3)
      GO TO 36
34 CALL PLOT(X, -14.2*3)
36 CALL PLOT(X, -14.2)
40 CONTINUE
CPLTF (Continued)

C * * DO LOOP 140 CALIBRATES THE BOTTOM NEGATIVE BORDER  
C * *  
  DO 140 I = 1,ITEMP  
     X = -I*SCALE1  
     IF(MOD(I,5).EQ.0) GO TO 132  
     CALL PLOT(X,-14,1,3)  
     GO TO 136  
  132 IF(MOD(I,10).EQ.0) GO TO 134  
     CALL PLOT(X,-14,16,3)  
     GO TO 136  
  134 CALL PLOT(X,-14,2,3)  
  136 CALL PLOT(X,-14,2)  
  140 CONTINUE  
C * * DO LOOP 240 CALIBRATES THE TOP POSITIVE BORDER  
C * *  
  DO 240 I = 1,ITEMP  
     X = I*SCALE1  
     IF(MOD(I,5).EQ.0) GO TO 232  
     CALL PLOT(X,1,3)  
     GO TO 236  
  232 IF(MOD(I,10).EQ.0) GO TO 234  
     CALL PLOT(X,16,3)  
     GO TO 236  
  234 CALL PLOT(X,2,3)  
  236 CALL PLOT(X,2)  
  240 CONTINUE  
C * * DO LOOP 340 CALIBRATES THE TOP NEGATIVE BORDER  
C * *  
  DO 340 I = 1,ITEMP  
     X = -I*SCALE1  
     IF(MOD(I,5).EQ.0) GO TO 332  
     CALL PLOT(X,1,3)  
     GO TO 336  
  332 IF(MOD(I,10).EQ.0) GO TO 334  
     CALL PLOT(X,16,3)  
     GO TO 336  
  334 CALL PLOT(X,2,3)  
  336 CALL PLOT(X,2)  
  340 CONTINUE  
C * * DO LOOP 200 CALIBRATES THE RIGHT HAND BORDER  
C * *  
  DO 200 I = 0,7C  
     Y = -14 + 0.2*I  
     IF(MOD(I,5).EQ.0) GO TO 16  
     CALL PLOT(1,-14,Y,3)  
     GO TO 18  

107
CPLTOF (Continued)

16 IF (MOD(I,16) .EQ. 0) GO TO 17
   CALL PLOT(16,Y,3)
   GO TO 18
17 CALL PLOT(16,Y,3)
18 CALL PLOT(16,Y,2)
20 CONTINUE
   AK1 = IC/IFULSCL
   CALL NUMBER(-0.06,-14.5,21,0,0,1)
   DO 200 I = 1,IFULSCL
   XPOS = -0.6 + I*AK1
   XNEG = -1.2 - I*AK1
   CALL NUMBER(XPOS,-14.5,21,1*IC,0,1)
   CALL NUMBER(XNEG,-14.5,21,1*IC,0,1)
   CALL SYMBOL(-4.5,-14.5,21,1)
   CALL SYMBOL(-4.5,-15.3,21,10,FCENTER,0,0)
   CALL NUMBER(-14.5,-15.3,21,9,IFULSCAL,0,0)
200 CONTINUE
   CALL PLOT(X,-1.0,3)
   CALL PLOT(X,Y,2)
   CALL PLOT(X,-1.0,2)
25 CONTINUE
   CALL PLOT(13,-2.0,3)
   CONTINUE
   CALL PLOT(13,-2.0,3)
   WRITE(6,36)
36 FORMAT(1HM*2X,14=PLOT COMPLETED)
   WRITE(6,36)
   CONTINUE
   RETURN
   END
SUBROUTINE FFT(A,IGAM,ISN)
COMPLEX A(1),T1,T2,TEMP
DOUBLE PRECISION PI2,SO,CO,SI,CI,SN,CS
PI2 = 6.28318530717958648D0
N = 2 ** IGAM
NBIT = 36 - IGAM
N1 = N - 2
DO 30 I = 1,N1
IFLIP = 0
IX = 1
DO 10 J = 1,IGAM
IOLD = IX
IX = IX / 2
IBIT = IOLD - 2 * IX
10 IFLIP = 2 * IFLIP + IBIT
IF (I .LE. IFLIP) GO TO 30
I1 = I + 1
I2 = IFLIP + 1
TEMP = A(I2)
A(I2) = A(I1)
A(I1) = TEMP
30 CONTINUE
DO 80 I = 1,IGAM
NEL = 2** I
NEL2 = NEL / 2
NSET = N / NEL
SI = DSIN(PI2/NEL)
CI = DCOS(PI2/NEL)
DO 80 J = 1,NSET
INCR = (J - 1) * NEL
SO = 0.*DD0
CO = 1.*DD0
DO 80 II = 1,NEL2
J1 = II + INCR
J2 = J1 + NEL2
T1 = A(J1)
T2 = A(J2) * CMPLX(CO, ISN * SO)
A(J1) = T1 + T2
A(J2) = T1 - T2
SN = SO * CI + CO * SI
CS = CO * CI - SO * SI
CO = CS
80 SO = SN
IF (ISN .GT. 0) GO TO 120
DO 110 I = 1,N
110 A(I) = A(I)/N
120 CONTINUE
RETURN
END
SUBROUTINE LFOLD (A,N)
COMPLEX A (1),T1
N2=N/2
DO 10 I=1,N2
II=I+N2
T1=A (I)
A (I)=A (II)
10 A (II)=T1
RETURN
END
SUBROUTINE PRNT(A,N)
COMPLEX A(I)
WRITE(6,101)
DO 10 I = 1,N
DB = 1.E30
T = CABS(A(I))
10 IF(T.GT.0.) DB = 20.* ALOG10(T)
WRITE(6,100) I,A(I),DB
100 FORMAT(IX,15,1P2EI5.,5X,OPFS,2)
101 FORMAT(//,- LINE-,?×,-REAL-,IIX,-IMAG-,12X,-DB-/)
RETURN
END

SUBROUTINE RDTF(NPNT,NPLOTS,TIME,PLOT)
DIMENSION TIME(NPNT), PLOT(NPNT,NPLOTS)
READ(19) TIME
READ(19) PLOT
RETURN
END

SUBROUTINE SCALE(_bMAX,MAXSCL)
C * * * THIS SUBROUTINE ESTABLISHES ORIGINATE SCALING FOR
C THE REMOTE SPECTRUM PLOTTER.
IF(_bMAX.LE.0.) GO TO 10
MAXSCL = 0
1 MAXSCL = MAXSCL + 10
DIFF = _bMAX - MAXSCL
IF(DIFF.GT.0.) GO TO 1
GO TO 999
10 MAXSCL = 0
11 MAXSCL = MAXSCL - 10
DIFF = _bMAX - MAXSCL
IF(DIFF.LE.0.) GO TO 11
MAXSCL = MAXSCL + 10
999 RETURN
END

111
SUBROUTINE TTFP(A,N,DELF,FLO,FHI)
C ** THIS SUBROUTINE PROVIDES A TELYTYPE PLOT OF THE FREQUENCY
C SPECTRUM FROM FLO TO FHI
COMPLEX A(1)
DIMENSION IA(50),MM(6)
C ** **
NST = (N/2) + INT(FLO/DELF + SIGN(.5,FLO))
NST = NST + 1
NSP = (N/2) + INT(FHI/DELF + SIGN(.5,FHI))
NSP = NSP + 1
C ** **
DBMAX = -1.E30
IEND = NSP - NST + 1
DO 1 I = 1,IEND
OCCTMP = CABS(A(NST + I - 1))
IF(DECTMP.LT.1.E-30) DECTMP = 1.E-30
B = 20. * ALOG10(DECTMP)
IF(B.GT.DBMAX) DBMAX = B
1 CONTINUE
C ** **
WRITE(6,2) IEND
2 FORMAT(13X,HNSIZE = ,15/)
C ** **
CALL SCALE(DBMAX,MAXSCL)
C ** **
The ordinate will vary from(MAXSCL-50) DB up
C TO MAXSCL DB.
DO 33 I = 1,50
33 IA(I) = I
DO 5 I = 1,6
5 MM(I) = MAXSCL - 10*(6-I)
MAXF = ABS(FLO)
IF(Abs(FHI).GT.MAXF) MAXF = ABS(FHI)
NAMEF = 0
IF(MAXF.GT.1.E3) NAMEF = 3
IF(MAXF.GT.1.E6) NAMEF = 6
IF(MAXF.GT.1.E9) NAMEF = 9
IF(NAMEF.EQ.0) WRITE(6,200)
IF(NAMEF.EQ.3) WRITE(6,203)
IF(NAMEF.EQ.6) WRITE(6,206)
IF(NAMEF.EQ.9) WRITE(6,209)
TTFP (Continued)

200 FORMAT(2X,14HFREQUENCY (HZ),8X,8HDECIBELS)
203 FORMAT(2X,15HFREQUENCY (KHZ),8X,8HDECIBELS)
206 FORMAT(2X,15HFREQUENCY (MHZ),8X,8HDECIBELS)
209 FORMAT(2X,15HFREQUENCY (GHZ),8X,8HDECIBELS)

WRITE(6,7) (MM(I),I = 1,6)
7 FORMAT(7X,14,A6X,5X,14)
WRITE(6,8)
8 FORMAT(9X,1HI,5(10H-+-+-+-+-))

FFACT = 1.
IF(NAMEF.EQ.3) FFACT = 1.E-3
IF(NAMEF.EQ.6) FFACT = 1.E-6
IF(NAMEF.EQ.9) FFACT = 1.E-9
FLO = FLO * FFACT
FHI = FHI * FFACT
DELF1 = DELF * FFACT
FLOPRT = DELF1*(NST -1 -N/2)
DO 10 I = 1,IEND
DECTMP = CABS(A(NST + 1 - I))
IF(DECTMP.LT.1.E-3) DECTMP = 1.E-3
B = 20. * ALOG10(DECTMP)
M = 50 + B - MAXSCL
J = I - 1
XJ = J
FREQ = FLOPRT + XJ * DELF1
IF(M.LT.U) GO TO 5U
IF(M.EQ.0) GO TO 34
DO 15 II = I,M
15 IA(II) = IA(I)
WRITE(6,35) FREQ,IA
DO 16 II = I,M
16 IA(II) = IA(I)
35 FORMAT(F8.3,2H 1,5UA1)
GO TO 36
34 WRITE(6,37) FREQ
37 FORMAT(1X,F7.3,2H -)
GO TO 36
5U WRITE(6,51) FREQ
51 FORMAT(1X,F7.3,2H 1)
1000 FORMAT( )
36 CONTINUE
10 CONTINUE
WRITE(6,*11)
11 FORMAT(6X,4HFREQ)
RETURN
SUBROUTINE TTP(A,NST,NSP,NJUMP)

COMPLEX A(I)

DIMENSION IA(50)

BMAX = 0.
BMIN = BMAX

DO 1 I = NST,NSP,NJUMP
  b = REAL(A(I))
  IF(b.LT.BMIN) BMIN = b
  IF(b.GT.BMAX) BMAX = b
  IF((BMAX-BMIN).LT.1.E-3) GO TO 999

1  DC 33 I = 1,50

33 IA(I) = IH

WRITE(6,4)

4 FORMAT(SX,IM+}

WRITE(6,3)

3 FORMAT(12X,-AMPLITUDE- MIN -,E9.4,- MAX -,E9.4,- VOLTS-)

WRITE(6,4)

WRITE(6,6)

6 FORMAT(8X,2H0,3X,2H1,3X,2H2,3X,2H3,3X,2H4,3X,2H5,3X,2H6,3X,2H7,3X,2H8,3X,2H9,3X,2H10,3X,2H11,3X,2H12,3X,2H13,3X,2H14,3X,2H15,3X,2H16,3X,2H17,3X,2H18,3X,2H19,3X,2H20,3X,2H21,3X,2H22,3X,2H23,3X,2H24,3X,2H25,3X,2H26,3X,2H27,3X,2H28,3X,2H29,3X,2H30,3X,2H31,3X,2H32,3X,2H33,3X,2H34,3X,2H35,3X,2H36,3X,2H37,3X,2H38,3X,2H39,3X,2H40,3X,2H41,3X,2H42,3X,2H43,3X,2H44,3X,2H45,3X,2H46,3X,2H47,3X,2H48,3X,2H49,3X,2H50)

WRITE(6,7)

7 FORMAT(1H0,8H1,1H1,1J(5H----1))

DO 10 I = NST,NSP,NJUMP
  b = REAL(A(I))
  b = (b-BMIN)/(BMAX-BMIN)
  M = INT(b*50.+0.5)
  IF(M.EQ.M) GO TO 34
  IA:M) = IH
  WRITE(6,35) I,IA
  IA:M) = IH

35 FORMAT(2X,15,3H I,SAl)

GO TO 36

34 WRITE(6,37) I

37 FORMAT(2X,15,3H *)

36 CONTINUE

10 CONTINUE

WRITE(6+11)

11 FORMAT(6X,1HN)

GO TO 900

999 WRITE (6+9)

9 FORMAT(1X,12HERRO RE FINISH)

900 RETURN

END
SUBROUTINE WRTDTA(NPNT,NPLOTS,TIME,PLOT)  
DIMENSION TIME(NPNT), PLOT(NPNT,NPLOTS)  
WRITE(6,110)  
READ(5,102) ISW  
IF(ISW GT 0) GO TO 10  
WRITE(6,101)  
WRITE(6,111) TIME  
WRITE(6,100) TIME  
GO TO 999  
10 IF(ISW GT NPLOTS) GO TO 20  
WRITE(6,101)  
WRITE(6,112) ISW  
WRITE(6,100) (PLOT(I,ISW), I = 1,NPNT)  
GO TO 999  
20 WRITE(6,113) NPLOTS  
100 FORMAT(1X,1P6E11.4)  
101 FORMAT(//)  
102 FORMAT( )  
110 FORMAT(- ENTER 0 FOR TIME LISTING OR DATA SET NUMBER-)  
111 FORMAT(3UX,-TIME-/)  
112 FORMAT(25X,-DATA SET NUMBER -12/)  
113 FORMAT(/- ERROR=-LARGEST DATA SET NUMBER IS -12/)  
999 WRITE(6,102)  
RETURN  
END
1. General Description

The program is coded in FORTRAN IV and the Univac-1108 version consists of a main program and 31 subroutines; in addition one of the subroutines (CPLOTF) which generates plots of frequency spectra on a CALCOMP plotter requires calls to 5 other subroutines contained in a plotter control package. This plotting routine and associated plotter control subroutines are not used in the SIGMA-5 version.

Several of the subroutines contain multiple entry points; the total number of subroutine and function entry names in the Univac-1108 version is 47. These names are listed in alphabetical order in Table C1 and those which are not subroutine names are identified.

In the following sections, the main program and all subroutines are briefly described, and each description is followed by a listing of the routine as used on the Univac-1108. For listings of the SIGMA-5 versions of those routines that were modified for that machine, see Appendix D.

NOTE: Many of the Hollerith strings in format statements were delimited with quote marks. The printer used to make the following listing did not have the quote character; a minus sign appears where each quote should have been.
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1. | ADJN |   |   |   |   |   |   | 25. | LFOUND |   |   |   |   |   |   | 26. | LIM |   |   |   |   |   |   |   |   |   |   |   |   | 118 |
| 2. | AMODEMO |   |   |   |   |   |   | 26. | LIM |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 3. | AMP |   |   |   |   |   |   | 27. | LSTCOM |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 4. | BCDFPT |   |   |   |   |   |   | 28. | NUMBER |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 5. | BWBNDP |   |   |   |   |   |   | 29. | PDCHK |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 6. | BWBSTP (BWBNDP) |   |   |   |   |   |   | 30. | PERIOD |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 7. | BWHIP (BWBNDP) |   |   |   |   |   |   | 31. | PHDEMO |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 8. | BWLOWP (BWBNDP) |   |   |   |   |   |   | 32. | PLOT |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 9. | CHBNDP |   |   |   |   |   |   | 33. | PLOTS |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|10. | CHBSTP (CHBNDP) |   |   |   |   |   |   | 34. | PROCES |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|11. | CHHIP (CHBNDP) |   |   |   |   |   |   | 35. | PRTFAC (PERIOD) |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|12. | CHLOWP (CHBNDP) |   |   |   |   |   |   | 36. | SCALE |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|13. | CPOLOTF |   |   |   |   |   |   | 37. | SIGGEN |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|14. | ELFIND |   |   |   |   |   |   | 38. | STRDTA |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|15. | FACTOR |   |   |   |   |   |   | 39. | SYMBOL |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|16. | FETCH |   |   |   |   |   |   | 40. | SYNBP |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|17. | FFT |   |   |   |   |   |   | 41. | SYNHNP (SYNRP) |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|18. | FILTER |   |   |   |   |   |   | 42. | SYNLNP (SYNBP) |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|19. | FLATSP |   |   |   |   |   |   | 43. | TELPLT |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|20. | FMDEMO |   |   |   |   |   |   | 44. | TIMFCN |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|21. | FRQFCN (TIMFCN) |   |   |   |   |   |   | 45. | TTPP |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|22. | FRQMUL |   |   |   |   |   |   | 46. | WRFF (WRTF) |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|23. | IDLMUL |   |   |   |   |   |   | 47. | WRTF |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|24. | INPFOR |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |

*Entry Point in subroutine named in parenthesis.

**Not used in SIGMA-5 version.

+CALCOMP plotter subroutines called by CPOLOT.
2. MAIN PROGRAM

Calls: PLOTS*, FETCH, ELFIND, WRTF, WRFF, TELPLT, TTFP, CPOFT, PRTFAC, STRDTA, PROCES, LISTCOM, INPFOR, FDCHK.

Commons: blank, CFREQ, CDOM, CDATA, CCIKRT, CWORD, CFLGS.

Description: MAIN is the overall controlling program which directs the operations of command and data input, interpretation of input, data storage, and command execution. Most of the detailed work in all operations is carried out by subroutines.

Program Listing:

```
PARAMETER NMAX = 2048
COMPLEX A(NMAX)
COMMON N,IGAM,DELF,DELT,PD,CARRFG
COMMON /CFREQ/ NFX,FR(6)
COMMON /CDUM/ DU,FLG
COMMON /CDATA/ JCTR,DATA(200)
COMMON /CCIKT/ NBLK,ITYP(3,2)
COMMON/CDORD/ WORD(10)
COMMON/CFGLS/ PDGLG,ARFLG
LOGICAL PDGLG, ARFLG
C DIMENSION IBUF(5000)
C CALL PLOTS(IBUF(1),5000,2)
JCTR = 1
ITYP(1,2) = JCTR
NFR = 0
NBLK = 0
IBLK = 0
PDGLG = .FALSE.
ARFLG = .FALSE.
WRITE(6,706)
1 DO 2 I = 1,10
2 WORD(I) = 6H
CALL FETCH(WORD,L,NBAD)
IF(NBAD .EQ. 0) GO TO 1
CALL ELFIND(WORD, LTYP)
GO TO( 10* 20, 30, 20, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200, 210, 220, 230, 240, 250, 260, 270, 280, 290, 300, 310, 320, 330, 340, 350, 360, 370, 380, 390, 400, 410, 420, 430) LTYP
```

*CALCOMP plotter routine.
MAIN (Continued)

10 IF (WORD(3) .EQ. 1H ) GO TO 12
   N1 = WORD(2)
   N2 = WORD(3)
   GO TO 15
12 WRITE(6*7004)
    READ(5*700U) N1,N2
15 CALL WRFF(A,N1,N2)
    GO TO 1
20 IF (WORD(3) .EQ. 1H ) GO TO 22
22 WRITE(6*7005)
    READ(5*700U) FRLO,FRHI
25 IF (LTYP .EQ. 4) GO TO 40
    CALL WRFF(A,FRLO,FRHI)
    GO TO 1
C * * TTY TIME PLOT
30 IF (WORD(3) .EQ. 1H ) GO TO 32
32 WRITE(6*7007)
    READ(5*700U) NST,NSP,NJUMP
35 IF (NJUMP .LT. 1) NJUMP = 1
    CALL TELPLT(A,NST,NSP,NJUMP)
    GO TO 1
C * * TTY FREQUENCY PLOT
40 CALL TTFP(A,FRLO,FRHI)
    GO TO 1
C * * CALCOMP TIME PLOT
50 WRITE(6*7101) ((ITYP(I,J)), J = 1*2, I = 1*5)
    7101 FORMAT(1X,2I3*3X))
    GO TO 650
C * * CALCOMP FREQUENCY PLOT
60 CALL CPlotF(A)
    GO TO 1
C * * PRINT PRIME FACTORS
70 CALL PRTFAC
    GO TO 1
C * * END OF JOB
80 GO TO 999
C ** BUTTERWORTH BANDPASS
 90 CALL STRDTA(3,0,0)
  NTYP = 3
  GO TO 600
C ** BUTTERWORTH LOWPASS
100 CALL STRDTA(2,0,0)
  NTYP = 4
  GO TO 600
C ** BUTTERWORTH HIGHPASS
110 CALL STRDTA(2,0,0)
  NTYP = 5
  GO TO 600
C ** BUTTERWORTH BANDSTOP
120 CALL STRDTA(3,0,0)
  NTYP = 6
  GO TO 600
C ** CHEBYSHEV BANDPASS
130 CALL STRDTA(4,0,0)
  NTYP = 7
  GO TO 600
C ** CHEBYSHEV LOWPASS
140 CALL STRDTA(3,0,0)
  NTYP = 8
  GO TO 600
C ** CHEBYSHEV HIGHPASS
150 CALL STRDTA(3,0,0)
  NTYP = 9
  GO TO 600
C ** CHEBYSHEV BANDSTOP
160 CALL STRDTA(4,0,0)
  NTYP = 10
  GO TO 600
C ** SYNCHRONOUS BANDPASS FILTER
170 CALL STRDTA(3,0,0)
  NTYP = 11
  GO TO 600
C ** SYNCHRONOUS LOWPASS FILTER
180 CALL STRDTA(2,0,0)
  NTYP = 12
  GO TO 600
C ** SYNCHRONOUS HIGHPASS FILTER
190 CALL STRDTA(2,0,0)
  NTYP = 13
  GO TO 600
C ** SIGNAL GENERATOR
200 CALL STRDATA(6,2,1)
   CARRFQ = WORD(2)
   NTYP = 1
   GO TO 600
C ** FREQUENCY MULTIPLIER
210 NTYP = 4
   GO TO 600
220 GO TO 430
C ** IDEAL MULTIPLIER
230 CALL STRDATA(2,1,2)
   NTYP = 16
   GO TO 600
240 GO TO 430
C ** FM DEMODULATOR
250 CALL STRDATA(1,0,0)
   NTYP = 19
   GO TO 600
C ** PHASE DEMODULATOR
260 CALL STRDATA(1,0,0)
   NTYP = 20
   GO TO 600
C ** AMPLIFIER
270 CALL STRDATA(1,0,0)
   NTYP = 2
   GO TO 600
C ** LIMITER
280 CALL STRDATA(3,0,0)
   NTYP = 17
   GO TO 600
290 NOUT = WORD(2)
   IF(NOUT .LE. NBLK) GO TO 291
   WRITE(6,7001) NBLK
   GO TO 1
291 IF(NOUT - IBLK) 295,295,293
293 IF(PDFLG) CALL PDCHK
   ITMP = IBLK + 1
   DO 292 IBLK = ITMP, NOUT
      IBTYP = ITYP(IBLK,1)
      JCTR = ITYP(IBLK,2)
      CALL PROCES(IBTYP,A)
   CONTINUE
   IBLK = NOUT
295 WRITE(6,7002) IBLK
   GO TO 1
MAIN (Continued)

300 CONTINUE
310 CONTINUE
320 GO TO 370
330 GO TO 430

C * * AM DEMODULATOR
340 CALL STRDTA(1,0,0)
   NTYP = 22
   GO TO 600

C * * FLAT SPECTRUM GENERATOR
350 CALL STRDTA(3,0,0)
   CARRFQ = 0
   NTYP = 23
   GO TO 600

360 CALL LSSTCOM
   GO TO 1
370 GO TO 650
380 CALL INPFOR
   GO TO 1
390 GO TO 650
400 GO TO 650
410 GO TO 650
420 GO TO 650
430 WRITE(6,7003)
   GO TO 1
600 NBLK = NBLK + 1
   ITYP(NBLK,1) = NTYP
   ITYP(NBLK + 1,2) = JCTR
   GO TO 1
650 WRITE(6,7100) WORD(1)
   GO TO 1

700 FORMAT(- * * ERROR * * LARGEST BLOCK NO IS -,12,- * *-)
701 FORMAT(- PROCESSING COMPLETE THRU BLOCK -,12)
702 FORMAT(- * * UNDEFINED STATEMENT * *-)
703 FORMAT(- ENTER LOW, HIGH INDICES-)
704 FORMAT(- ENTER LOW, HIGH FREQUENCIES-)
705 FORMAT(- START-)
706 FORMAT(- ENTER NSTART, NSTOP, NJUMP-)
710 FORMAT(- COMMAND -,AS-, IS NOT YET OPERATIONAL-)
999 CONTINUE

C CALL PLOTS(U*,0,,999)
STOP
END
3. Subroutine ADJN

Called by: PDCHK, PERIOD, FLATSP

Calls: none

Commons: blank

Entries: none

Description: ADJN adjusts N (number of data samples) to be a power of 2. If the current value of N (either that computed to meet the Nyquist criterion or that entered by the user) is already a power of 2 it is not changed; otherwise it is adjusted upward to the next power of 2.

Program Listing:

```
SUBROUTINE ADJN
COMMON N,IGAM,DELF,DELT,PD
IGAM = ALOG(N)/ALOG(2.) + .999
N = 2**IGAM
DELT = PD / N
RETURN
END
```
4. Subroutine AMDEMO

Called by: PROCES
Calls: FRQFCN, TIMFCN
Commons: blank, CDATA
Entries: none

Description: AMDEMO simulates an ideal amplitude demodulator. Operating on the frequency spectrum, the negative frequency components are all set
to zero to give a spectrum characteristic of a complex time function.
The positive frequency components are then shifted down in the data array
by an amount corresponding to the center frequency of the demodulator,
thus positioning the spectrum at baseband. Transforming to the time
domain produces a complex time waveform; conversion to a real time waveform is effected by replacing each time sample with one whose real part
is the absolute value of the complex sample, and whose imaginary part is
set to zero.

Program Listing:

SUBROUTINE AMDEMO(A)
COMMON N*IGAM*DELF*DELT*PD*CAHRFU
COMMON/JCTR*,DATA(200)/
COMPLEX A(I)
PI2 = 6.2831853
N2 = N/2
F0 = WORD(JCTR)
CALL FRQFCN(A)

C ** REMOVE THE NEGATIVE FREQUENCY COMPONENTS
DO 10 I = 1+N2
10 A(I) = (0.,0.)

C ** MOVE THE MODULATED CARRIER TO ZERO FREQUENCY
IFU = F0 / DELF + .5
NSTART = N2 + 1
NSTOPI = N - IFU + 1
DO 11 I = NSTART,N
11 A(I - IFU) = A(I)

DO 12 I = NSTOPL,N
12 A(I) = (0.,0.)

C ** RECOVER THE AMPLITUDE INFORMATION
CALL TIMFCN(A)
DO 20 I = 1,N
TEMP = CABSI(A(I))
20 A(I) = CMPLX(TEMP,0.)
RETURN
END
5. Subroutine AMP

Called by: PROCES
Calls: none
Commons: blank, CDATA
Entries: none

Description: AMP simulates an amplifier. Its action is simply to multiply each data sample by a constant. Since the multiplication is the same in both time and frequency domains, AMP accepts the data array in either domain.

Program Listing:

```
SUBROUTINE AMP(A)
COMPLEX A(I)
COMMON N,IGAM
COMMON/CDATA/ JCTR,DATA(200)
XP = DATA(JCTR) / 200.
G = 10.**XP
DO 100 I = 1,N
100 A(I) = G*A(I)
RETURN
END
```
6. Function BCDFTP

Called by: FETCH

Calls: none

Commons: none

Entries: none

Description: BCDFTP accepts binary coded characters representing numerical quantities and converts them to a real number which is returned through the function name. This routine was adapted from CIRCUS with a few minor changes.

Program Listing:

```
FUNCTION BCDFTP (C, N)

BCDFTP CONVERTS DATA FROM BCD TO FLOATING POINT.
C  IS AN ARRAY CONTAINING THE NUMERIC CHARACTERS
WHICH ARE TO BE CONVERTED.
I  = INDEX OF THE CHARACTER BEING CONVERTED.
J  = INDEX CORRESPONDING TO THE DIGIT J-1.
K  = 1 WHEN DECODING WHOLE NUMBER PORTION.
2 WHEN DECODING FRACTIONAL PORTION.
3 WHEN DECODING EXPONENT.
INTEGER DIGIT, F, PLUS, DECPT, PCD
LOGICAL EXPFLG, DIGFLG, DECFLG, EXSIGN
DIMENSION BCD(4), KSIGN(2), NTGR(3), RESULT(3), DIGIT(12)
DATA DIGIT / 1H, 1H, 1H, 1H, 1H, 1H, 1H, 1H, 1H, 1H, 1H, 1H,
DATA PLUS, MINUS, E, DECPT / 1H, 1H, 1H, 1H, 1H, 1H,
EXPFLG = .FALSE.
DIGFLG = .FALSE.
DECFLG = .FALSE.
EXSIGN = .FALSE.
DO 11 K = 1, 3
    KSIGN(K) = 1
    NTGR(K) = 1
11 CONTINUE
APLART = 0
K = 1
DO 31 I = 1, N
    CHAR = BCD(I)
```
FUNCTION BCDFPT (Continued)

C* TEST FOR SIGN, DIGIT, DECIMAL POINT, OR E
C*
13 IF ( ICHAR-PLUS ) 13,23,13
14 GO TO 15
15 CONTINUE
16 IF ( ICHAR-DECPNT ) 16,26,16

PLUS SIGN

23 IF ( DIGFLG ) GO TO 28
GO TO 31

MINUS SIGN

24 IF ( DIGFLG ) GO TO 27
SIGN(1) = -1
GO TO 31

DIGIT FROM 0 TO 9

25 INTEGR(K) = 10*INTEGR(K)+J-1
NPLART = NPLART+K-1
DIGFLG = *TRUE*
GO TO 31

DECIMAL POINT

26 IF ( DECFLG ) GO TO 21
IF ( EXPFLG ) GO TO 21
DECFLG = *TRUE*
K = 2
GO TO 31

128
FUNCTION BCDFTP (Continued)

C*  
C*  BLANK TIMES TF, ** EXPONENT NOT ALLOWED.
C*  
C*  
27 KSIGN(3) = -1
28 IF ( EXSIGN ) GO TO 21
   EXSIGN = .TRUE.
   GO TO 33;
29 IF ( EXPFLG ) GO TO 21
   IF ( .NOT. DIGFLG ) GO TO 21
30, EXPFLG = .TRUE.
31 CONTINUE
C*  THE NUMBER HAS BEEN SEPARATED INTO INTEGER, FRACTION,
C*  AND EXPONENT PARTS. COMBINE THEM TO FORM THE
C*  NUMBER IN FLOATING POINT.
C*  
32 IF ( EXPFLG ) GO TO 32
33 EXPON = 1.
   GO TO 35
C*  
C*  CALCULATE EXPONENT. AN EXPONENT MAY BE ONLY TWO
C*  DIGITS LONG AND LESS THAN 38 IN MAGNITUDE.
C*  
32 IF ( NPLART-NPLASV-4 ) 33,33,21
33 IEXPON = INTEGR(3)*KSIGN(3)
   IF ( IABS( IEXPON ) - 37 ) 34,34,21
34 EXPON = 10.**IEXPON
   NPLART = NPLASV
C*  
C*  CALCULATE MANTISSA
C*  
35 RTSHFT = 1.**NPLART
   RESULT(1) = FLOAT( INTEGR(1)*KSIGN(1) )
   RESULT(2) = FLOAT( INTEGR(2)*KSIGN(1) ) / RTSHFT
   BCDFTP = ( RESULT(1)+RESULT(2) )**EXPON
41 RETURN
C*  
C*  ILEGSAL CHARACTER OR BAD SYNTAX.
C*  
21 N = -1
   GO TO 41
END
7. Subroutine BWBNDP

Called by: PROCES
Calls: FILTER
Commons: CDATA, CFILT
Entries: BWBSTP, BLOWP, BWHID

Description: BWBNDP simulates a Butterworth bandpass filter; auxiliary entries produce simulations of Butterworth band stop, high pass, and low pass filters.

The Butterworth filter produces a maximally flat response defined by the function

\[ |H(\omega_p)| = \frac{1}{\sqrt{1 + (\omega_p^2)^n}} \]  

(1)

where \( n \) is the filter order and \( \omega_p \) is a normalized frequency. The poles of Equation (1) are given by

\[ S_k = \frac{1}{2\pi} \arccos \left( \frac{\omega^2}{\omega_p^2} \right) \]  

(2)

where

\[ \theta = \frac{(2k + n - 1)}{n} \frac{\pi}{2}, \quad k = 1, 2, \ldots, 2n. \]  

(3)

With the poles known, the transfer function can be written in terms of the \( n \) poles lying in the left half-plane as

\[ H(s) = \frac{1}{(s - S_1)(s - S_2)(s - S_3) \ldots (s - S_n)} \]  

(4)

The Butterworth filter models are implemented with Equations (2), (3), and (4). The computation of the transfer function, Equation (4), is carried out in subroutine FILTER. Subroutine BWBNDP computes the values of \( S_k \), and sets up variables from which the normalized frequency, \( \omega_p \), can be determined to satisfy the definitions

\[ \omega_p = \omega/\omega_c \] for low pass,

\[ \omega_p = \omega - \omega_0 \] for band pass,
\[ \omega_p = \omega / \omega_c \text{ for high pass,} \]
\[ B_w = 2 \frac{\omega}{\omega - \omega_0} \text{ for band stop,} \]

where

\[ \omega_c = 3 \text{ dB corner frequency, and} \]
\[ B_w = \text{full 3 dB bandwidth.} \]

All Butterworth filters calculated with this model will exhibit 3 dB attenuation at the corner frequency (low pass and high pass), or at one-half the bandwidth away from the center frequency (band pass and band stop). The computed transfer function of the latter two are symmetrical.

Program Listing:

```fortran
SUBROUTINE BW_BNDP(A)
COMPLEX A(I),S(20)
COMMON/CDATA/ JCTR,DATA(20)
COMMON/CFILT/ F0,FCOFF,RF,AMP,FFLG,S
LOGICAL FFLG
FFLG = .TRUE.
GO TO 2
ENTRY BW_BANDP(A)
FFLG = .FALSE.
2 FO = DATA(JCTR)
FCOFF = DATA(JCTR+1) / 2.
NR = DATA(JCTR+2)
GO TO 6
ENTRY BW_LOWP(A)
FFLG = .TRUE.
GO TO 4
ENTRY BW_HIP(A)
FFLG = .FALSE.
4 FO = 0.
FCOFF = DATA(JCTR)
NR = DATA(JCTR+1)
6 AMP = 1.
DO 10 K = 1,NR
THETA = 1.5707963 * ((2*K + NR-1)/NR)
10 S(K) = CMPLX(COS(THETA), SIN(THETA))
CALL FILTER(A)
RETURN
END
```
8. Subroutine CHBNDP

Called by: PROCES
Calls: FILTER
Commons: CDATA, CFILT
Entries: CHBSTP, CHLOWP, CHHIP

Description: CHBNDP simulates a Tchebysheff bandpass filter; auxiliary entries produce simulations of Tchebysheff bandstop, high pass, and low pass filters.

The implementation of the Tchebysheff (equal ripple) filter model is identical to that used for Butterworth filters except for the computation of the poles. The poles for the Tchebysheff filter are given by

\[ S_k = \sigma_k + j\omega_k \]  \hspace{1cm} (1)

where

\[ \sigma_k = \pm \tanh a \sin \theta, \]
\[ \omega_k = \cos \theta \]
\[ a = \frac{1}{n} \sinh^{-1} \frac{1}{\varepsilon}, \]
\[ \theta = \left( \frac{2k - 1}{n} \right) \left( \frac{n}{2} \right), \quad k = 1, 2, 3, \ldots 2n, \]
\[ \varepsilon = \text{ripple width}, \quad 0 < \varepsilon < 1. \]

CHBNDP computes the poles and then calls subroutine FILTER which actually computes the transfer function and applies it to the frequency function.
Program Listing:

SUBROUTINE CHBNDP(A)
COMPLEX A(1),S(20)
COMMON/CDATA/ JCTR,DATA(200)
COMMON/CFILT/ F0,FCOFF,NR,AMP,FFLG,S
LOGICAL FFLG
FFLG = .TRUE.
GO TO 2
ENTRY CHBSTP(A)
FFLG = .FALSE.
2 FU = DATA(JCTR)
FCOFF = DATA(JCTR+1) / 2.
NR = DATA(JCTR+2)
EPSD= = DATA(JCTR+3)
GO TO 6
ENTRY CHLOWP(A)
FFLG = .TRUE.
GO TO 4
ENTRY CHHIP(A)
FFLG = .FALSE.
4 FU = 0.
FCOFF = DATA(JCTR)
NR = DATA(JCTR+1)
EPSD= = DATA(JCTR+2)
6 X = 1. / SQRT(EXP(+23.25851 * EPSDB) - 1.)
ARG = X + SQRT(X ** 2 + 1)
AE = ALOG(ARG) / NR
CALL FQFNCN(A)
DO 10 K = 1,NR
THETA = 1.5707963 * ((2.*(K+NR)-1)/NR)
SIGK = TANH(AE) * SIN(THETA)
OMEGK = COS(THETA)
10 S(K) = CMPLX(SIGK,OMEGK)
FFAC = COSH(AE)
IF (.NOT. FFLG) FFAC = 1./FFAC
FCOFF = FFAC * FCOFF
AMP = 1.
DO 15 K = 1,NR
15 AMP = AMP * CADBS(S(K))
IF (MOD(NR+2) .EQ. 0) AMP = AMP / EXP(+11512925*EPSDB)
CALL FILTER(A)
RETURN
END
9. Subroutine CPLOTF

Called by: MAIN

Calls: FRQFCN, SCALE, FACTOR*, PLOT*, NUMBER*, SYMBOL*

Commons: blank

Entries: none

Description: CPLOTF is used to produce high quality plots of frequency spectra. The routine actually generates a data file suitable for driving an off-line CALCOMP plotter. Since the routine embodies both equipment and procedural considerations, its use is probably limited to the Univac-1108 and CALCOMP plotter at Georgia Tech. It is included here for completeness.

Program Listing:

```
C * *
SUBROUTINE CPLOTF(A)
C ** THE FOLLOWING CONTROL STATEMENT MUST PRECEDE THE EXECUTE STATEMENT FOR RUNS USING CALCOMP PLOTTERS.
C **
C ** @USE UNIT #, TPS$
C **
C *
COMMON N,ICA,DEL$
COMPLEX A(1)
CALL FRQFCN(A)
1799 FORMAT(I)
WRITE(6,1716)
1716 FORMAT(- ENTER FLO AND FHI FOR CALCOMP SPECTRUM PLOT. -)
READ(5,1799) FLO,FHI
XTEST = ABS(FHI-FLO)
IF(XTEST.LT.1.E-3) GO TO 9999
XTEST = XTEST/ABS(FLO) + ABS(FHI))
IF(XTEST.LT.1.E-3) GO TO 9999
YSPRED = 7..*
N2 = N/2
NST = N2 + INT(FLO/DELFL) + 1
NSP = N2 + INT(FHI/DELFL) - 1
T1 = 1.E-35
DC 4000 I = NST+NSP
T2 = UABS(A(I))
4000 IF(T2.GT.T1) T1 = T2
UMAX = 20.*ALOG1(T1)
CALL SCALE(UMAX,MAXSCL)
MXSCL = MAXSCL
MXSCL = 10.**(-XVXSCL/2.**)
```

*CALCOMP plotter routines
CALL FACTOR(6,4)
CALL PLOT(6,6,2,3)
CALL PLOT(12,6,0,3)
CALL PLOT(-1,14,3)
DO 3 I = 1,2
 CALL PLOT(-1,2,1,2)
 CALL PLOT(1,6,2,2)
 CALL PLOT(1,14,2)
 CALL PLOT(-1,31,1,2)
 CALL PLOT(1,31,1,2)
 CALL PLOT(1,1,1,2)
 CALL PLOT(1,14,1,2)
3 CALL PLOT(-1,14,1,2)
 DO 3 10 I AGAIN = 1,2
 DO 1, I = 1,73
 Y = -14* + 1.2*I
 IF (Y (MOD(I,5) .EQ. 1)) GO TO 6
 CALL PLOT(-16,1,Y,3)
 GO TO 8
6 IF (Y(1,1) .EQ. 1) GO TO 7
 CALL PLOT(-16,16,Y,3)
 GO TO 8
7 CALL PLOT(-16,2,Y,3)
 CALL PLOT(-16,2,Y,2)
10 CONTINUE
 Y = 2*
 DO 15 I = 1,8
 J = I - 1
 Y = Y-2
 YY = XYSCL = 1.*J#YSPRED/76.
15 CALL NUMBER(-11.99,Y-1.5*21,YY+(-1)
 CALL SYMBOL(-11.24,-8.4*21,14;AMPLITUDE (D9),90.0,14)
 FCENTR = (FLC + FH])/2
 FUPPER = FH - FCENTR
 IF(FUPPERGT.1*2) GO TO 2100
 IEXP = ALG1(FUPPER) = 1
 GO TO 2101
2100 CONTINUE
 XIEXP = ALG1 (FUPPER)
 IEXP = XIEXP
 RIEXP = IEXP
 IF(140.XISXIEXP-RIEXP)2LT.1.E-2) .AND. (XIEXP.GE.RIEXP)
 I 1 .IERP = 1
 CONTINUE
2101 CONTINUE
 FULSCL = FUPPER*(1.**(1-IXP))
 IFLSCL = FULSCL
 ITEMP = 1.*IFLSCL
 XIT:VP = ITEMP
 SCAL:1 = !IFLSCL
CPLOTF (Continued)

SCALE1 = 1.0*SCALE1/FULSCL
TEINF1 = 1.0*IFL.SC1
SCALEF1 = SCALE1/TEINF1
GO TO 40

I = ITMP
X = I*SCALE1
IF (MOD(I,5) .NE. 0) GO TO 32
CALL PLOT(X,-14,1,3)
GO TO 36

32 IF (MOD(I,10) .EQ. 0) GO TO 34
CALL PLOT(X,-14,16,3)
GO TO 36

34 CALL PLOT(X,-14,2,3)
36 CALL PLOT(X,-14,2)

40 CONTINUE
DO 140 I = I,ITMP
X = I*SCALE1
IF (MOD(I,5) .EQ. 0) GO TO 132
CALL PLOT(X,-14,1,3)
GO TO 136

132 IF (MOD(I,10) .EQ. 0) GO TO 134
CALL PLOT(X,-14,16,3)
GO TO 136

134 CALL PLOT(X,-14,2,3)
136 CALL PLOT(X,-14,2)

140 CONTINUE
DO 240 I = I,ITMP
X = I*SCALE1
IF (MOD(I,5) .EQ. 0) GO TO 232
CALL PLOT(X,-1,3)
GO TO 236

232 IF (MOD(I,10) .EQ. 0) GO TO 234
CALL PLOT (X,-16,3)
GO TO 236

234 CALL PLOT(X,-2,3)
236 CALL PLOT(X,-2)

240 CONTINUE
DO 340 I = I,ITMP
X = I*SCALE1
IF (MOD(I,5) .EQ. 0) GO TO 332
CALL PLOT(X,1,3)
GO TO 336

332 IF (MOD(I,10) .EQ. 0) GO TO 334
CALL PLOT(X,16,3)
GO TO 336

334 CALL PLOT(X,2,3)
336 CALL PLOT(X,2)
340 CONTINUE
CPLTF (Continued)

DO 2 I = 1, 7
Y = -14 - 2 * I
IF (Y-PLUS(I,5) .GE. 0) GO TO 16
CALL PLOT (1, Y, 2)
GO TO 14
15 IF (Y-Minus(I,5) .GE. 0) GO TO 17
CALL PLOT (1, Y-16, 2)
GO TO 14
17 CALL PLOT (1, Y+4, 2)
18 CALL PLOT (1, Y+2, 2)
20 CONTINUE
41 = 1 * FULLCL
CALL FILLER(-14.5, 14.5, 21, 1, 1, 1)
DO 2 I = 1, 1 * FULLCL
XPOS = -0.5 + I * 4.5
XPOS = -0.5 - I * 4.5
CALL SYMBOL(XPOS, -14.5, 14.5, 21, 1, 1, 1, 10)
2 CALL SYMBOL(-20.5, -15.5, 21, 1, 1, 1, 10)
CALL SYMBOL(0.5, -15.5, 21, 1, 1, 1, 10)
CALL SYMBOL(20.5, -15.5, 21, 1, 1, 1, 10)
3 CALL SYM (4.5, 14.5, 21, 1, 1, 1, 10)
31 CONTINUE
ISTOP = ISTOP - 1
VSET = VSET - 1
DELAY = 2000
J = 1, 1, 1
TI = (5 * (14.5 - 7.) / VPOS)
IF (TI .LT. 1.) GO TO 3
Y1 = (7. * (15.5 - 4.5) * 4.) * ALOG10(TI)
IF (Y1 .LT. -14.5) GO TO 3
X1 = 1. * DELAY - 1.
IF (X1 .LT. -15.5) GO TO 3
IF (X1 .LT. 1.) GO TO 3.
IF (Y1 .LT. -15.5) GO TO 25
CALL SYM (Y1, 15.5, 21, 1, 1, 1)
GO TO 3
25 CONTINUE
CALL PLOT(X1, -14.5, 2)
CALL PLOT(X1, Y1, 2)
CALL PLOT(X1, 14.5, 2)
3 CONTINUE
CALL PLOT(15.5, -2, 1)
36 FORMAT (14X, 12X, "PLOT COMPLETE")
WRITE (6, 33)
999 CONTINUE
RETURN
END

137
10. Subroutine ELFIN

Called by: INPFOR, MAIN, PDCHK

Calls: none
Commons: none
Entries: none

Description: ELFIN compares a Hollerith string of up to six characters to a number of pre-stored character strings. When a match is found, an integer is set to a unique value which indicates the matched string. This is the basic operation of identifying the input commands; the integer is returned to the calling program and used to direct program flow to properly execute the command. This subroutine was patterned after a similar subroutine in CIRCUS, but is essentially a complete rewritten version.

Program Listing:

```
SUBROUTINE ELFIN (NAME, L)
PARAMETER NMAX = 42

ELFIN TRIES TO MATCH THE KEY WORD FROM AN INPUT DATA STRING (NAME) AGAINST ONE OF THE ALLOWABLE INPUT FORMS.
L IS SET TO THE INDEX WHICH CORRESPONDS TO THE MATCHED INPUT TYPE.

DIMENSION MATCH(NMAX)
DATA (MATCH(1), I=1,NMAX)
A /*HPRINTT, HPRINTF, LTPLODT, LTPLOTF, LHCPLOTT, LHCPLOTF*/
B /*HPRIMEF, HPRIMED, 3H3X3NP, 3H3X3NP, 633X3HIP, 633X3PSTP*/
C /*6HCHDNP, 6HCHELNP, 6HCHELIP, 6HCHELSTP, 5HSYBP, 5HSHYLP*/
D /*6HSHYBP, 6HSIGEF, 6HFRMUL, 6H6GIDLML, 6H*/
E /*6HFLDEMO, 6HFLDEMO, 3HAMP, 3HLIM, 6HLOCK, 3HYF, 3F*/
F /*2HNO, 1HN, 6H*/
G /*6HCIRCUI, 6HINPUTF, 6HDELETE, 6HINSERT, 6HREPLOC, 6HP$DEAT*/
DO 11 I = 1, NMAX
IF (NAME = MATCH(I)) 11,21,11
11 CONTINUE
I = NMAX + 1
21 L = I
RETURN
END
```
11. Subroutine FETCH

Called by: MAIN, PDCHK
Calls: BCDPT
Commons: CFETCH
Entries: none

Description: FETCH is the main input routine, it reads in commands as a string of BCD characters, decodes the various elements in the input stream and stores them in array WORD. All blank characters are discarded; different elements are delimited by commas. Hollerith strings are truncated to the first six characters and stored in WORD. Numeric characters representing data are converted to real numbers by BCDPT prior to storage. FETCH was also adapted from CIRCUS, but several changes were made. In particular, the program was modified to eliminate two calls to assembly language subroutines.

```
SUBROUTINE FETCH(A POST, LL, BA D)
INTEGER APOST, BLANK, COMMA, DEPT, EQUAL, PLUS, PO PAREN, TEXT
INTEGER SUFF1, SUFF2, SUFF, ADOPT, TITLE, WORD, E
DIMENSION TIT LE(12)
COMMON/CFETCH/ SUFF2(6), SUFF1(50)
DIMENSION WORD(1)
DATA APOST, BLANK, COMMA, DEPT, EQUAL, PLUS, TITLE

1 / 1H-, 1H+, 1H-, 1H0, 1H0, 1H+ /
DATA LPAREN, RPAREN
1 / 1H(, 1H) /
DATA E / 1HE /
END

NCOLS = 87
READ = 1

FETCH IS A FREE-FIELD INPUT SUBPROGRAM WHICH READS
THE INPUT DATA IN LL CONSECUTIVE CELLS OF THE
ARRAY WORD. HOLLERTH IS TRUNCATED TO 6 CHAR.
CONTINUE
READ (5*1, 1, END = 1, 7) (SU FF1(I), I = 1, 50)
WRITE (5*1, 1, 5) (SU FF1(I), I = 1, 50)
1 Y = .
2 \"X = .
NCOMMA = .

139
```
FETCH (Continued)

DO 3 I = 1, 6
3 BBB2(I) = BLANK

C 4 IF ( M = NCOLS ) 40, 160, 160
C * EXAMINE EACH COLUMN, REMOVE BLANKS, AND TEST FOR
C SEPARATORS.
C 40 N = N + 1
C TEST = BUFF1(M)
C IF ( TEST = BLANK ) 41, 4, 41
C 41 IF ( TEST = COMMA ) 42, 4, 42
C 42 IF ( TEST = EQUAL ) 43, 4, 43
C 43 IF ( TEST = LPAREN ) 44, 4, 44
C 44 IF ( TEST = RPAREN ) 45, 4, 45
C 45 N = N + 1
C BBB2(N) = TEST
C IF ( K ) 5, 5, 4
C * IF TYPE HAS NOT BEEN SET (K=0), TEST CHARACTER TO
C DETERMINE IF IT IS A DIGIT OR SIGN (NO DECISION),
C A 4-8 PUNCH (TITLE CARD), A DECIMAL POINT (FLOATING
C POINT NUMBER), OR NONE OF THESE, IN WHICH CASE A
C HOLLERITH WORD IS ASSUMED. IF K IS SET, IT WILL BE
C 0 WHEN AN INTEGER
C 1 WHEN A FLOATING POINT NUMBER
C 2 WHEN A HOLLERITH WORD.
C 5 NCOMMA = 1
C 51 IF ( TEST = NZ ) 52, 51, 51
C 52 IF ( TEST = NINE ) 4, 4, 52
C 53 IF ( TEST = PLUS ) 53, 4, 53
C 54 IF ( TEST = MINUS ) 54, 4, 54
C 55 IF ( TEST = APOST ) 55, 4, 55
C 56 IF ( TEST = DECP ) 56, 57, 56
C 57 IF ( TEST = E ) 561, 561, 561
C 560 IF ( N-1 ) 561, 561, 4
C 561 K = 2
C GO TO 4
C 57 K = 1
C GO TO 4
FETCH (Continued)

C * SELECT MODE OF CONVERSION BASED UPON K.
6 IF ( K<1 ) 7,7,7
7 SUFF3 = SUFF3P( SUFF2*N )
IF ( N ) 1,6,106,3
8 WORD(L+1) = SUFF3
GO TO 91
9 ENCOP$(N,11,13,WORD(L+1)) ( SUFF2(1),1=1,6 )
91 L=L+1
C * IF NOT FINISHED WITH THE CARD IMAGE, REINITIALIZE
C AND CONTINUE. IF THE NCOLS COLUMN CONTAINED A
C COMMA, PROCESS THE NEXT CARD. OTHERWISE, SET THE
C NUMBER OF WORDS CONVERTED IN LL AND RETURN.
C IF ( N-NCOLS+2*1+1 ) 1,1,11
1 IF ( ( N Test = COMMA ) 11,1,11
11 LL = L
RETURN
C * MOVE A TITLE CARD INTO THE TITLE ARRAY.
3 ENCOP$(72,13,1,TITLE) ( SUFF1(1),1 = 1,72 )
GO TO 1
C * A CARD IMAGE HAS BEEN PROCESSED. IF THE LAST
C NON-BLANK SYMBOL WAS A COMMA (NCOMMA=1), READ
C THE NEXT CARD. OTHERWISE THERE IS INFORMATION
C IN *SUFF3* TO BE CONVERTED, AFTER WHICH, FETCH
C WILL RETURN TO THE CALLING PROGRAM.
C IF ( ( NCOMMA ) 6,1,6
C * FETCH FOUND CONVOLUTED SEPARATORS OR A NUMBER WITH
C MORE THAN 15 DIGITS AND COULD HAD NOT CONTINUE.
1,6 WRITE (6,20) ( SUFF2(1),1=1,6 )
WHAD = 7
RETURN
1,7 STOP
1001 FORMAT(8,4,51)
1002 FORMAT(6,4,1)
1005 FORMAT(1X8,4,1)
2005 FORMAT(25H1) FCPH-- CANNOT CONVOP 6A1,44 -- // //)
END
12. Subroutine FFT

Called by: TIMFCN

Calls: none
Commons: none
Entries: none

Description: FFT performs the direct and inverse fast Fourier transform. This program is substantially the same FFT routine developed under Contract NASA8-20054 and previously reported*. It has been modified, however, to remove the FLD function, available in FORTRAN V, which appeared in the original version. These changes appear in the DO 10 loop, and the version listed here contains only standard FORTRAN-IV statements.

Program Listing:

```
SUBROUTINE FFT(A,IGAM,ISN)
COMPLEX A(1:1),T1,T2,TEMP
DOUBLE PRECISION P12,SO,CO,SI,CI,SN,CS
P12 = 6.28315397179586480D0
N = 2 ** IGAM
NBIT = 36 - IGAM
N1 = N - 2
DO 30 I = 1,N1
IFLIP = 0
IX = I
DO 10 J = 1,IGAM
IOLD = IX
IX = IX / 2
IBIT = IOLD - 2 * IX
10 IFLIP = 2 * IFLIP + IBIT
IF (I.IE.1,IFLIP) GO TO 30
I1 = I + 1
I2 = IFLIP + 1
TEMP = A(I2)
A(I2) = A(I1)
A(I1) = TEMP
30 CONTINUE
DO 80 I = 1,IGAM
NEL = 2** I
NEL2 = NEL / 2
NSET = N / NEL
SI = DSIN(P12/NEL)
CI = DCOS(P12/NEL)
DO 80 J = 1,NSET
INCR = ( J - 1 ) * NEL
SO = 0.*DDO
CO = 1.*DDO
DO 80 II = 1,NEL2
J1 = II + INCR
J2 = J1 + NEL2
T1 = A(J1)
T2 = A(J2) * CMPLX(CO, ISN * SO)
A(J1) = T1 + T2
A(J2) = T1 - T2
SN = SO * CI + CO * SI
CS = CO * CI - SO * SI
CO = CS
80 SO = SN
IF (ISN .GT. 0) GO TO 120
DO 110 I = 1,N
110 A(I) = A(I)/N
120 CONTINUE
RETURN
END
```
13. Subroutine FILTER

Called by: BWBNDP, CHBNDP

Calls: FRQFCN

Commons: blank, CFILT

Entries: none

Description: FILTER operates on the components in the frequency array to complete the computations for any type of Butterworth or Tchebysheff filter. Given the poles, $S_k$, determined by BWBNDP or CHBNDP, FILTER calculates the transfer function

$$H(f_p) = \frac{1}{(f_p - S_1)(f_p - S_2) \cdots (f_p - S_n)}$$

where $n$ is the filter order (number of poles) and $f_p$ is a normalized complex frequency defined by

$$f_p = \begin{cases} 
\frac{f - f_o}{f_{\text{cutoff}}} & \text{low pass band pass filters,} \\
\frac{f_{\text{cutoff}}}{f - f_o} & \text{high pass or band stop filters.}
\end{cases}$$

All spectral lines subject to more than 300 dB rejection are set to zero.

Program Listing:

```fortran
SUBROUTINE FILTER(A)
COMPLEX A(I:20),M,Z
COMMON N,IAMO,DELFT,DELFT
COMMON/CFILT/ F0,FCOFF,N1,AMP,FLG,S
LOGICAL FLG
TEST = EXP(35.* / NV)
CALL FRQFCN(A)
DO 30 I = 1,N
II = I - 1 - N/2
F = II * DELF
FP = SIGN(1.*F) * (ABS(F) - F0) / FCOFF
IF (FLG) GO TO 15
IF (ABS(FP) .LT. 1.E-16) GO TO 25
FP = -1.*FP
15 IF (ABS(FP) .GT. TEST) GO TO 25
Z = CMPLX(0.,FP)
HD = CMPLX(1.,0.)
DO 20 K = 1,N1
20 A(K) = HD * (Z - S(K))
A(I) = AMP * A(I) / HD
GO TO 30
25 A(I) = CMPLX(0.,0.)
30 CONTINUE
RETURN
END
```

144
14. Subroutine FLATSP

Called by: PROCES
Calls: ADJN
Commons: blank, CDATA, CDOM
Entries: none
Description: FLATSP loads the frequency array with components of uniform amplitude thus simulating the spectrum of an impulse function. It is useful in examining the transfer functions of filters in detail.

Program Listing:

```fortran
SUBROUTINE FLATSP(A)
  COMPLEX A(I)
  COMMON N,IGAM,DELF,DELT,PD
  COMMON /CDATA/ JCTR,DATA(20)
  COMMON /CDOM/ DOMFLG
  LOGICAL DOMFLG
  AMP = DATA(JCTR)
  DELF = DATA(JCTR+1)
  PD = 1./DELF
  N = DATA(JCTR+2)
  CALL ADJN
  DO 10 I = 1,N
    10 A(I) = CMPLX(AMP,0.)
    DOMFLG = .FALSE.
    RETURN
END
```
15. Subroutine FMDEMO

Called by: PROCES

Calls: FRQFCN, TIMFCN

Commons: blank, CDATA, CDOM

Entries: none

Description: FMDEMO simulates the action of an FM demodulator. Operating on the frequency spectrum, the negative frequency components are all set to zero to give a spectrum characteristic of a complex time function. The positive frequency components are then shifted down in the data array by an amount corresponding to the center frequency of the demodulator, thus positioning the spectrum at baseband. Transforming to the time domain and taking the complex logarithm of each time sample produces an imaginary part equal to the phase angle (modulo $2\pi$). A tracking loop corrects for excursions beyond the $\pm \pi$ range thus reconstructing the phase deviation due to the angle modulation.
Program Listing:

```
SUBROUTINE FMDEMOIA)
COMMON N*1GAM, DELT, PD, CARRFQ
COMMON/CDATA/ JCTR, DATA(2U0)
COMPLEX A(1)
PI2 = 6.2831853
/
N2 = N /2
F0 = WORD(JCTR)
CALL FRQFCN(A)
C * * REMOVE THE NEGATIVE FREQUENCY COMPONENTS
DO 10 I = 1, N2
10 A(I) = (0., 0.)
C * * MOVE THE MODULATED CARRIER TO ZERO FREQUENCY
IFU = F0 / DELT + .5
NSTART = N2 + 1
NSTOPI = N - IFG + 1
DO 11 I = NSTART, N
11 A(I - IFU) = A(I)
DO 12 I = NSTOPI, N
12 A(I) = (0., 0.)
C * * RECOVER THE ANGLE INFORMATION
CALL TIXFCN(A)
A(1) = CLOG(A(1))
THETA2 = AIMAG(A(1))
DO 20 I = 2, N
A(I) = CLOG(A(I))
THETA1 = AIMAG(A(I))
THETAT = THETA1 * THETA2
IF(THETAT .LE, 0,) SC TO 21
GO TO 29
21 IF(ABS(THETA1) .LE, 1.57) GO TO 29
IF(THETA2 .GE, 0.) GO TO 22
J = J - 1
GO TO 29
22 J = J + 1
29 THETA2 = THETA1
TEMP = THETA1 + PI2 * J
20 A(I) = CMPLX(TEMP, 0.)
A(I) = CMPLX(AIMAG(A(I)), 0.)
C * * ZERO THE D-C COMPONENT
CALL FRQFCN(A)
A(NZ + 1) = CMPLX(0., 0.)
RETURN
END
```
16. Subroutine FRQMUL

Called by: PROCES
Calls: TIMFCN
Commons: blank
Entries: none

Description: FRQMUL provides the action of a biased half-wave rectifier; operating on the time function, it passes only those time samples whose amplitude exceed a fixed threshold (currently set at 0.5 volts). The resulting signal is rich in harmonics of the carrier frequency. Particular multiples can be isolated by filtering. Note that the output of FRQMUL is not bandlimited and the user should be aware that aliasing may be a problem.

Program Listing:

```
SUBROUTINE FRQMUL(A)
  DIMENSION A(1)
  COMMON N, IGAM
  CALL TIMFCN(A)
  THRES = .5
  NDOL = 2 * N - 1
  DO 100 I = 1, NDOL, 2
  A(I) = A(1) - THRES
  A(I + 1) = 0.
  IF (A(I) .LT. 0.) A(I) = 0.
100 CONTINUE
RETURN
END
```
17. Subroutine IDLMUL

Called by: PROCES
Calls: TIMFCN
Commons: blank, CDATA, CFREQ
Entries: none
Description: IDLMUL is an ideal multiplier which operates in the time domain and generates the product of the signal being processed and a local oscillator signal.

Program Listing:

```
SUBROUTINE IDLMUL(A)
COMMON N*IGAM*DELF*DELT
COMMON/CDATA/ JCTR*DATA(20)
COMMON/CFREQ/ NFR*FR(6)
COMPLEX A(1)
AMPLO = DATA(JCTR)
FLO = DATA(JCTR + 1)
CALL TIMFCN(A)
PI2 = 6.2831d53
WLO = PI2 * FLO
DO 1 I = 1,N
II = I - 1
T = II * DELT
1   A(I) = A(I) * AMPLO * SIN(WLO * T)
RETURN
END
```
18. Subroutine INPFOR

Called by: MAIN

Calls: ELFIND

Commons: CWORD

Entries: none

Description: INPFOR is a service routine that will list the input format and define the parameters of any block input command. It is added as a convenience to the remote terminal user; it has no affect on the circuit or signal being processed.

Program Listing:

```
SUBROUTINE INPFOR
COMMON /CWORD/ WORD(16)
LOGICAL FLG
FLG = .TRUE.
CALL ELFIND(WORD(2),L)
GO TO ( 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 9, 10,
        A  11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
        d  21, 1, 23, 1, 25, 26, 27, 28, 1, 1,
        C  1, 1, 1, 34, 35, 1, 1, 1, 1, 1,
        D  1, 1, 560), L
1 WRITE(6,7004) WORD(2)
   GO TO 999
20 WRITE(6,7020)
    WRITE(6,7120)
    IF (FLG) GO TO 999
35 WRITE(6,7035)
    WRITE(6,7135)
    IF (FLG) GO TO 999
9 WRITE(6,7009)
    WRITE(6,7109)
    IF (FLG) GO TO 999
10 WRITE(6,7010)
    WRITE(6,7110)
    IF (FLG) GO TO 999
11 WRITE(6,7011)
    WRITE(6,7111)
    IF (FLG) GO TO 999
12 WRITE(6,7012)
    WRITE(6,7112)
    IF (FLG) GO TO 999
13 WRITE(6,7013)
    WRITE(6,7113)
    WRITE(6,7113)
    IF (FLG) GO TO 999
```

150
INPFOR (Continued)

14 WRITE (6, 71.14)
15 WRITE (6, 71.21)
16 WRITE (6, 71.31)
17 IF (FLG) GO TO 999
18 WRITE (6, 71.15)
19 WRITE (6, 71.22)
20 WRITE (6, 71.32)
21 IF (FLG) GO TO 999
22 WRITE (6, 71.16)
23 WRITE (6, 71.23)
24 WRITE (6, 71.33)
25 IF (FLG) GO TO 999
26 WRITE (6, 70.17)
27 WRITE (6, 70.24)
28 WRITE (6, 70.17)
29 IF (FLG) GO TO 999
30 WRITE (6, 70.18)
31 WRITE (6, 70.25)
32 WRITE (6, 70.25)
33 IF (FLG) GO TO 999
34 WRITE (6, 70.19)
35 WRITE (6, 70.26)
36 WRITE (6, 70.19)
37 IF (FLG) GO TO 999
38 WRITE (6, 70.27)
39 WRITE (6, 70.27)
40 IF (FLG) GO TO 999
41 WRITE (6, 70.28)
42 WRITE (6, 71.10)
43 IF (ORD (?, FLG, 3-ILL)) GO TO 910
44 WRITE (6, 71.11) ORD (?,)
45 GO TO 999
46 FLG = FALSE,
47 GO TO 2.
INPFOR (Continued)

7000 FORMAT()
7001 FORMAT(1X,A6,- IS NOT AN INPUT COMMAND-)
7002 FORMAT(// BWENDP, FC, BW, NR-)
7003 FORMAT(// BLOWP, FC, NR-)
7004 FORMAT(// BVHP, FC, NR-)
7005 FORMAT(// CHBNDP, FC, BW, NR, EPSDB-)
7006 FORMAT(// CHEWP, FC, NR, EPSDB-)
7007 FORMAT(// CHHIP, FC, NR, EPSDB-)
7008 FORMAT(// CHBSTP, FC, BW, NR, EPSDB-)
7009 FORMAT(// CHBSTP, FC, BW, NR, EPSDB-)
7010 FORMAT(// FXAB, FC, NR-)
7011 FORMAT(// FXAB, FC, NR-)
7012 FORMAT(// FXAB, FC, NR-)
7013 FORMAT(// FXAB, FC, NR-)
7014 FORMAT(// FXAB, FC, NR-)
7015 FORMAT(// FXAB, FC, NR-)
7016 FORMAT(// FXAB, FC, NR-)
7017 FORMAT(// FXAB, FC, NR-)
7018 FORMAT(// FXAB, FC, NR-)
7019 FORMAT(// FXAB, FC, NR-)
7020 FORMAT(// FXAB, FC, NR-)
7021 FORMAT(// FXAB, FC, NR-)
7022 FORMAT(// FXAB, FC, NR-)
7023 FORMAT(// FXAB, FC, NR-)
7024 FORMAT(// FXAB, FC, NR-)
7025 FORMAT(// FXAB, FC, NR-)
7026 FORMAT(// FXAB, FC, NR-)
7027 FORMAT(// AMP, GAIN/- GAIN = VOLTAGE GAIN, DB-)
    A = (6 DB = FACTOR OF 2)-
7028 FORMAT(// LIN, CL, CH, GL-)
7029 FORMAT(// AMDEVC, FC, NR-)
7030 FORMAT(// FLATSP, AMP, DELF, N-)
7031 FORMAT(// FLATSP, AMP, DELF, N-)
7032 FORMAT(// FLATSP, AMP, DELF, N-)
7033 FORMAT(// FLATSP, AMP, DELF, N-)
7034 FORMAT(// FLATSP, AMP, DELF, N-)
7035 FORMAT(// FLATSP, AMP, DELF, N-)
7036 FORMAT(// FLATSP, AMP, DELF, N-)
7037 FORMAT(// FLATSP, AMP, DELF, N-)
7038 FORMAT(// FLATSP, AMP, DELF, N-)
7039 FORMAT(// FLATSP, AMP, DELF, N-)
7040 FORMAT(// FLATSP, AMP, DELF, N-)
7041 FORMAT(// FLATSP, AMP, DELF, N-)
7042 FORMAT(// FLATSP, AMP, DELF, N-)
7043 FORMAT(// FLATSP, AMP, DELF, N-)
7044 FORMAT(// FLATSP, AMP, DELF, N-)
7045 FORMAT(// FLATSP, AMP, DELF, N-)
7046 FORMAT(// FLATSP, AMP, DELF, N-)
7047 FORMAT(// FLATSP, AMP, DELF, N-)
7048 FORMAT(// FLATSP, AMP, DELF, N-)
7049 FORMAT(// FLATSP, AMP, DELF, N-)
7050 FORMAT(// FLATSP, AMP, DELF, N-)
7051 FORMAT(// FLATSP, AMP, DELF, N-)
7052 FORMAT(// FLATSP, AMP, DELF, N-)
7053 FORMAT(// FLATSP, AMP, DELF, N-)
7054 FORMAT(// FLATSP, AMP, DELF, N-)
7055 FORMAT(// FLATSP, AMP, DELF, N-)
7056 FORMAT(// FLATSP, AMP, DELF, N-)
7057 FORMAT(// FLATSP, AMP, DELF, N-)
7058 FORMAT(// FLATSP, AMP, DELF, N-)
7059 FORMAT(// FLATSP, AMP, DELF, N-)
7060 FORMAT(// FLATSP, AMP, DELF, N-)
7061 FORMAT(// FLATSP, AMP, DELF, N-)
7062 FORMAT(// FLATSP, AMP, DELF, N-)
7063 FORMAT(// FLATSP, AMP, DELF, N-)
7064 FORMAT(// FLATSP, AMP, DELF, N-)
7065 FORMAT(// FLATSP, AMP, DELF, N-)
7066 FORMAT(// FLATSP, AMP, DELF, N-)
7067 FORMAT(// FLATSP, AMP, DELF, N-)
7068 FORMAT(// FLATSP, AMP, DELF, N-)
7069 FORMAT(// FLATSP, AMP, DELF, N-)
7070 FORMAT(// FLATSP, AMP, DELF, N-)
7071 FORMAT(// FLATSP, AMP, DELF, N-)
7072 FORMAT(// FLATSP, AMP, DELF, N-)
7073 FORMAT(// FLATSP, AMP, DELF, N-)
7074 FORMAT(// FLATSP, AMP, DELF, N-)
7075 FORMAT(// FLATSP, AMP, DELF, N-)
7076 FORMAT(// FLATSP, AMP, DELF, N-)
7077 FORMAT(// FLATSP, AMP, DELF, N-)
7078 FORMAT(// FLATSP, AMP, DELF, N-)
7079 FORMAT(// FLATSP, AMP, DELF, N-)
7080 FORMAT(// FLATSP, AMP, DELF, N-)
7081 FORMAT(// FLATSP, AMP, DELF, N-)
7082 FORMAT(// FLATSP, AMP, DELF, N-)
7083 FORMAT(// FLATSP, AMP, DELF, N-)
7084 FORMAT(// FLATSP, AMP, DELF, N-)
7085 FORMAT(// FLATSP, AMP, DELF, N-)
7086 FORMAT(// FLATSP, AMP, DELF, N-)
7087 FORMAT(// FLATSP, AMP, DELF, N-)
7088 FORMAT(// FLATSP, AMP, DELF, N-)
7089 FORMAT(// FLATSP, AMP, DELF, N-)
7090 FORMAT(// FLATSP, AMP, DELF, N-)
7091 FORMAT(// FLATSP, AMP, DELF, N-)
7092 FORMAT(// FLATSP, AMP, DELF, N-)
7093 FORMAT(// FLATSP, AMP, DELF, N-)
7094 FORMAT(// FLATSP, AMP, DELF, N-)
7095 FORMAT(// FLATSP, AMP, DELF, N-)
7096 FORMAT(// FLATSP, AMP, DELF, N-)
7097 FORMAT(// FLATSP, AMP, DELF, N-)
7098 FORMAT(// FLATSP, AMP, DELF, N-)
7099 FORMAT(// FLATSP, AMP, DELF, N-)
8000 WRITE(6,7000)
8001 RETURN
8002 END
19. Subroutine LFOLD

Called by: TIMFCN
Calls: none
Commons: none
Entries: none

Description: LFOLD provides the action of folding and unfolding the frequency spectrum to meet the requirements of the FFT. The frequency domain representation is always ordered by frequency except when entering or leaving the FFT.

Program Listing:

```
SUBROUTINE LFOLD (A,N)
COMPLEX A (1), T1
N2=N/2
DO 10 I=1,N2
  II=I+N2
  T1=A (I)
  A (I)=T1
  A (II)=A (II)
10  A (II)=T1
RETURN
END
```
20. Subroutine LIM

Called by: PROCES
Calls: TIMFCN
Commons: blank, CDATA
Entries: none

Description: LIM operates on the time function to produce ideal limiting. Signal excursions are clipped to specified upper and lower limit levels. Note that the output of LIM is not bandlimited and the user should be aware that aliasing may be a problem.

Program Listing:

SUBROUTINE LIM(A)
COMMON N
COMMON/CDATA/ JCTR,DATA(200)
CLEVL = DATA(JCTR)
CLEVH = DATA(JCTR+1)
GL = DATA(JCTR+2)
COMPLEX A(I)
CALL TIMFCN(A)
DO 1 I = 1,N
  A(I) = GL*A(I)
  IF (REAL(A(I)) .LE. CLEVL) A(I) = CMPLX(CLEVL,0.0)
  IF (REAL(A(I)) .GE. CLEVH) A(I) = CMPLX(CLEVH,0.0)
1 RETURN
END
21. Subroutine LSTCOM

Called by: MAIN

Calls: none

Commons: CWORD

Entries: none

Description: LSTCOM is a service routine that will list all of the valid commands that are recognized by FATCAT. It is included as an aid to the remote terminal user; calling LSTCOM has no affect on the circuit or signal being processed.

Program Listing:

```
SUBROUTINE LSTCOM
COMMON XCORRY, WORD(1)
WRITE(5,7,1)
I = 1
IF (.NOT. WORD(2).EQ. 8).GO TO 20
    I = 0
IF (.NOT. WORD(2).EQ. 9).GO TO 20
    I = 0
IF (.NOT. WORD(2).EQ. 6).GO TO 20
    I = 0
IF (.NOT. WORD(2).EQ. 7).GO TO 20
    I = 0
IF (.NOT. WORD(2).EQ. 3).GO TO 20
    I = 0
IF (.NOT. WORD(2).EQ. 2).GO TO 20
    I = 0
IF (.NOT. WORD(2).EQ. 1).GO TO 20
    I = 0
WRITE(6,7,2) WORD(2)
GO TO 999
7   WRITE(6,7,2)
   IF (I.EQ. 1) GO TO 999
3   WRITE(6,7,3)
   IF (I.EQ. 2) GO TO 999
4   WRITE(6,7,4)
   IF (I.EQ. 3) GO TO 999
5   WRITE(6,7,5)
   IF (I.EQ. 4) GO TO 999
6   WRITE(6,7,6)
   IF (I.EQ. 5) GO TO 999
7   WRITE(6,7,7)
   GO TO 999
8   WRITE(6,7,8)
999 RETURN
7,01 FORMAT (15X=DATA COMMAND SUMMARY=)
7,02 FORMAT (1X=SOURCE//--SIGN--*5X=SIGN SIGNAL GENERATOR--/
A=FLATSP--5X=FLAT SPECTRUM GENERATOR--)
```
LSTCOM (Continued)

7003 FORMAT(/,10X,-FILTERS/- BUTTERWORTH/-)
   A - 3WNDP-5X,-BAND PASS/-
   B - BLOWP-5X,-LOW PASS/-
   C - BWHP-6X,-HIGI PASS/-
   D - BWPST-5X,-BAND STOP/-
   E - CHEBYSHEFF/-
   F - CHNPD-5X,-BAND PASS/-
   G - CHLOWP-5X,-LOW PASS/-
   H - CHHP-6X,-HIGH PASS/-
   I - CHSTP-5X,-BAND STOP/-
   J - SYNCHRONOUSLY TUNED/-
   K - SYNB-6X,-BAND PASS/-
   L - SYNL-6X,-LOW PASS/-
   M - SYNHP-6X,-HIGH PASS/-

7004 FORMAT(/,10X,-DEMODULATORS/-)
   A - FMDEMO-5X,-FM DEMODULATOR/-
   H - AMDEMO-5X,-AM DEMODULATOR/-
   C - PMDEMO-5X,-PHASE DEMODULATORS/-

7005 FORMAT(/,10X,-MISCELLANEOUS/-)
   A - FROMUL-5X,-FREQUENCY MULTIPLIER/-
   C - IDLMUL-5X,-IDEAL MULTIPLIER/-
   D - AMP-8X,-AMPLIFIER/-
   E - LIM-8X,-LIMITER/-

7006 FORMAT(/,10X,-CONTROL COMMANDS/-)
   A - PRINTT-5X,-PRINT TIME FUNCTION/-
   B - PRINTF-5X,-PRINT FREQUENCY FUNCTION/-
   C - TPLOTT-5X,-PRINTER PLOT OF TIME FUNCTION/-
   D - TPLOTF-5X,-PRINTER PLOT OF FREQUENCY FUNCTION/-
   E - CPLOTT-5X,-REMOTE PLOT OF TIME FUNCTION/-
   F - CPLOTF-5X,-REMOTE PLOT OF FREQUENCY FUNCTION/-
   G - BLOCK-6X,-PROCESS SIGNAL TO OUTPUT OF BLOCK SPECIFIED/-
   H - PRIMEF-5X,-LIST PRIME FACTORS OF ALL SOURCE FREQUENCIES/-
   I - END OF JOB-1X,-TERMINATES RUN/-
   J - INPUT FORMATS,BLOCK NAME LIST INPUTS FOR NAMED BLOCK/-

7007 FORMAT(/,4X,-SPECIAL COMMANDS TO SPECIFY ARRAY SIZE/-)
   A - YES-8X,-LISTED ARRAY SIZE ACCEPTABLE/-
   B - NO-9X,-LISTED ARRAY SIZE NOT ACCEPTABLE/-
   C - N-10X,-SET ARRAY SIZE TO SPECIFIER VALUE/-

7008 FORMAT/- LIST COMMANDS- COMMANDS ARE LISTED BY GIVING-,
   A - TWO ALPHANUMERIC/- WORD SETS SEPARATED BY A COMMA-,
   B - THE FIRST IS -1H-=-LIST COMMAND-1H/- POSSIBLE-,
   C - SECOND WORDS AND THE RESULTING OUTPUTS ARE-,
   D - ALL-12X,-LIST ENTIRE COMMAND SET-,
   E - SOURCES-8X,-LIST COMMANDS FOR SOURCE BLOCKS-,
   F - FILTERS-8X,-LIST COMMANDS FOR FILTERS-,
   G - DEMODULATORS-3X,-LIST COMMANDS FOR DEMODULATORS-,
   H - MISC-11X,-LIST OTHER BLOCK COMMANDS (AMP, LIM, ETC)-,
   I - CONTROL-8X,-LIST CONTROL COMMANDS-,
   J - NSIZE-12X,-LIST COMMANDS CONTROLLING ARRAY SIZE-,
   K - LIST COMMANDS-2X,-LIST THE ABOVE INFORMATION/-

7020 FORMAT/- LSTCOM CANNOT DECODE --A6/-
END
22. Subroutine PDCHK

Called by: MAIN

Calls: PERIOD, FETCH, ELFIN D, ADJN, PRIFAC

Commons: blank, CWORD, CFLGS

Entries: none

Description: PDCHK is used in checking the period of the data set and ascertaining that the Nyquist criterion is met. This subroutine is largely executive in nature; the major calculations are carried out by subroutine PERIOD. PDCHK is called when processing is called for, provided a new source frequency has been added since the last processing call. On its initial call, it determines the period of the data set, and the associated Δf, Δt, and array size. On any subsequent calls (which happen only if a new source frequency has been introduced), it checks to see if all frequencies are still periodic on the established period. If they are, processing continues; if not, an error message is written and the run terminated.

Program Listing:

```
SUBROUTINE PDCHK
COMMON N, IGAM, DFL, DFLT, PD
COMMON /CWORD/ WORO(1)
COMMON /CFLGS/ DFLGS, AREFLG
LOGICAL DFLGS, AREFLG

IFLAG = 0
NSET = 0
CALL PERIOD
IF(ARFLG) GO TO 50
NSET = 4
WRITE(6,7) PD, DFL
1. WRITE(6,7 1) N, IGAM, DFLT
2. CALL FETCH(CWORD, N, IGAM)
   CALL ELFIN D(CWORD, LTYP)
   IF(LTYP =EQ. 3) GO TO 100
   IF(LTYP =EQ. 9) GO TO 30
   IF(LTYP =EQ. 91) GO TO 40
   IF(LTYP =EQ. 81) STOP
   WRITE(6,7 2)
   GO TO 2
```

157
PDCHK (Continued)

30 WRITE(6,7003)
   GO TO 20
40 N = WORD(2)
    NFLG = 0
    CALL ADJN
    IF(N .GE. NSFT) GO TO 10
    NFLG = 1
    GO TO 900
50 IF(NSFT .GE. N) GO TO 990
    WRITE(6,7005) N
    CALL PRTFAC
    WRITE(6,7006)
    STOP
900 IF(NFLG .EQ. 0) GO TO 999
    WRITE(6,7004) N,NSFT
    WRITE(6,7003)
    GO TO 20
999 N = NSFT
999 NFLG = .FALSE.
   ARFLG = .TRUE.
   RETURN
7000 FORMAT(/,-- PERIOD = -,1PE10.3,-- SECONDS,--,DELTA-F = -,A
     E10.3)
7001 FORMAT(-- N = -,16,-- IGAM = -,12,-- DELTA-T = --
      1,1PE10.3/- IS THIS SATISFACTORY--)
7002 FORMAT(-- INPUT MEANINGLESS * ENTER YES, NO, OR N'S_VALUE--)
7003 FORMAT(= ENTER N, VALUE--)
7004 FORMAT(= N = -,16,-- UNACCEPTABLE ** N MUST--
      1 -- BE -,16,-- TO MEET NYQUIST CRITFRION--)
7005 FORMAT(= SOURCE FREQUENCIES REQUIRE THE ARRAY SIZE --
      1 -- TO BE -,19// PRIME FACTORS ARE--)
7006 FORMAT(=/-- RUN IS BEING TERMINATED--) END
23. Subroutine PERIOD

Called by: PDCHK
Calls: ADJN
Commons: blank, CFREQ
Entries: PRTFAC

Description: PERIOD factors each source frequency into prime factors and constructs the highest common factor to determine Δf and the associated smallest period on which all of the frequencies are periodic. From the period and the highest frequency present, the number of samples to meet the Nyquist criterion is computed; this number is adjusted upward (if necessary) to a power of 2 by ADJN.

Entry PRTFAC will produce a listing of the prime factors of the source frequencies.

Program Listing:

```fortran
SOURCUT PEOID PERIOD EXAMINES NEXT FREQUENCIES (MAX 6) IN ARAY F.
FACORS EACH INTO PRIME FACTORS, CONSTRUCTS DELF AS THE
GREAST COMMON FACTOR COMPURES ARAY SIZE N AND ICA.
TO SATISFY THE NYQUIST CRITERION AND FFT REQUIREMENTS.
COMPURE THE TOTAL PERIOD PD AND SAMPLING PERIOD DELT.
COMMON N, ICA, PD, DELT, P
COMMON/CFREQV/FNERY(6)
DIMENSION IA(30,6), IOUT(30), ICT(6)
LOGICAL FLAG
FLAG = .TRUE.
GO TO 5
ENTRY PRTFAC
FLAG = .FALSE.
```

159
PERIOD (Continued)

5  MAXFAC = 0
   DO 100 I = 1,NFR
C ** CONVERT ITH FREQ TO INTEGER.
   IFR = FRI(I)
C ** CLEAR ITH COL OF FACTOR ARRAY IA.*
   DO 10 J = 1,30
10  IA(J,I) = 0
   J = 1
   ITST = 2
   IDEL = 1
C ** FACTOR ITH FREQ INTO PRIME FACTORS.
   GO TO 40
30  ITST = ITST + IDEL
   IDEL = 2
40  IF(IFR .EQ. 1) GO TO 90
   ALIV = SQRT(IFR)
   IF(ITST .GT. ALIV) GO TO 60
50  IF(IMOD(IFR,ITST) .NE. 0) GO TO 30
   IA(J,I) = ITST
   J = J + 1
   IF(J .GE. 30) GO TO 90
   IFR = IFR / ITST
   GO TO 50
60  IA(J,I) = IFR
90  IF (J .GT. MAXFAC) MAXFAC = J
100 CONTINUE
   IF (FLAG) GO TO 142
   LIM = MAXFAC + 1
   DO 101 I = 1,LIM
101  WRITE(6,711) I,(IA(I,JI,J)*1,NFR)
   GO TO 999
C ** ALL FREQS FACTORED. PRIME FACTORS IN FIRST NFR COLS.
C ** OF IA. FIND COMMON FACTORS AND PLACE IN IOUT.
102  IOCT = 1
   DO 110 I = 1,NFR
110  ICT(I) = 1
120  IMAX = 0
   DO 130 I = 1,NFR
130  IF(IA(I,1) .GT. IMAX) IMAX = IA(I,1)
135  DO 200 I = 1,NFR
140  J = ICT(I)
   IF(IA(J,1) .EQ. 0) GO TO 400
   IF(IA(J,1) .GE. IMAX) GO TO 200
   ICT(I) = J + 1
   GO TO 140
200  CONTINUE
PERIOD (Continued)

C * * IF ALL POINT FACTORS ARE NOT IMAX, ADVANCE IMAX.
21. DO 22 I = 1, NFR
       J = ICT(I)
       IF(A(J, I) .NE. IMAX) GO TO 391
22. CONTINUE
C * * ALL FACTORS ARE IMAX, TRANSFER TO IOUT, ADVANCE ALL
C * * POINT FACTORS, STEP, AND RESTART FOR COMMON FACTOR.
       IOUT(I, ICT) = IMAX
       ICT = ICT + 1
       IF (ICT .GE. NFR) GO TO 20
       ICT(I) = ICT(I) + 1
       GO TO 391
23. IMAX = IMAX + I
       GO TO 135
24. ICT = ICT - 1
       SELF = 1.
C * * IS ANY FACTOR IN IOUT,SELF = 1. OTHERWISE SELF = 0.
C * * PRODUCT OF ALL PRIMES IN ICT.
       IF(IOUT(I, ICT) .NE. IC) GO TO 415
       DO 40 I = 1, ICT
60. SELF = SELF*10
       IOUT(I) = IC(I)
40   CONTINUE
41. PD = 1. /SELF
       FM(X) = 0.
       DO 42 I = 1, NFR
     42. IF(FR(I) .LE. FM(X)) FM(X) = FR(I)
C * * SAMPLING PERIOD TO MEET MAXIMUM CRITERION.
C * * P = 2. * PD * FMAX + .5
C * * ARRAY SIZE AND SAMPLE PERIOD (DELT) TO SATISFY PRT.
       CALL ADJF
       GO TO 999
99. WRITE(6,7) 1
     7. STOP
     1. FORMAT(//10, *FREQUENCY = ,12, *HAS MORE THAN 29 FACTORS =/)
711. FORMAT(1X, A(1X, 16))
99 RETURN
END
24. Subroutine PHDEMO

Called by: PROCES
Calls: FRQFCN, TIMFCN
Commons: blank, CDATA
Entries: none

Description: PHDEMO simulates the action of an ideal phase demodulator. Operating on the frequency spectrum, the negative frequency components are all set to zero to give a spectrum characteristic of a complex time function. The positive frequency components are then shifted down in the data array by an amount corresponding to the center frequency of the demodulator, thus positioning the spectrum at baseband. Transforming to the time domain and taking the complex logarithm of each time sample produces an imaginary part equal to the phase angle (modulo 2π). A tracking loop corrects for excursions beyond the ±π range thus reconstructing the phase deviation due to the angle modulation.
SUBROUTINE PhDEMO(A)
COMMON N,IGAM,DELF,DELT,P0,CARRFQ
COMMON/CDATA/ JCTR,DATA(2JG)
COMPLEX A(1)
PI2 = 6.2831853
J = 0
N2 = N / 2
FO = WORD(JCTR)
CALL FRQFCN(A)
C ** REMOVE THE NEGATIVE FREQUENCY COMPONENTS
DO 10 I = 1,N2
10 A(I) = (U*, U*)
C ** MOVE THE MODULATED CARRIER TO ZERO FREQUENCY
IFU = FO / DELF + .5
NSTART = N2 + 1
NSTOP1 = N - IFU + 1
DO 11 I = NSTART,N
11 A(I - IFU) = A(I)
DO 12 I = NSTOP1,N
12 A(I) = (U*, U*)
C ** RECOVER THE ANGLE INFORMATION
CALL TIMFCN(A)
A(1) = CLOG(A(1))
THETA2 = AIMAG(A(1))
DO 20 I = 2,N
A(I) = CLOG(A(I))
THETA1 = AIMAG(A(I))
THETAT = THETA1 - THETA2
IF(THETAT LE. 0.) GO TO 21
GO TO 29
21 IF(ABS(THETA1) LE. 1.57) GO TO 29
IF(THETA2 GE. 0.) GO TO 22
J = J - 1
GO TO 29
22 J = J + 1
29 THETA2 = THETA1
TEMP = THETA1 + PI2 * J
20 A(I) = CMPLX(TEMP, U*)
A(1) = CMPLX(AIMAG(A(1)), U*)
C ** ZERO THE D-C COMPONENT
CALL FRQFCN(A)
A(N2 + 1) = (0*, 0*)
RETURN
END
25. Subroutine PROCES

Called by: MAIN

Calls: SIGGEN, AMP, BWBNDP, BWLOWP, BWHIP, BWBSTP, CHBNDP, CHLOWP, CHIP, CHBSTP, SYNB, SYNP, SYNH, FRQMUL, IDLMUL, LIM, FMDEMO, PHDEMO, AMDEMO, FLATSP.

Commons: none

Entries: none

Description: PROCES is simply a switching routine that calls the proper block model for processing the signal.
SUBROUTINE PROCESS(I, A)

COMPLEX A(I)

GO TO ( 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
1       11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
2       21, 22, 23), I

1 CALL SIGGEM(A)
RETURN
2 CALL AMP(A)
RETURN
3 CALL BNDP(A)
RETURN
4 CALL BLOWP(A)
RETURN
5 CALL BWHIP(A)
RETURN
6 CALL BWBRTP(A)
RETURN
7 CALL CHNDP(A)
RETURN
8 CALL CHLOWP(A)
RETURN
9 CALL CHHIP(A)
RETURN
10 CALL CHDSTP(A)
RETURN
11 CALL SYNJP(A)
RETURN
12 CALL SYNLG(A)
RETURN
13 CALL SYNHLP(A)
RETURN
14 CALL FRGMUL(A)
RETURN
15 CALL IDLMUL(A)
RETURN
16 CALL LIM(A)
RETURN
19 CALL FMDEMUt(A)
RETURN
20 CALL PHDEMU(A)
RETURN
21 CALL AMDemo(A)
RETURN
22 CALL FLATSPE(A)
RETURN
END
26. Subroutine SCALE

Called by: CPLOTF, TTFP

Calls: none

Commons: none

Entries: none

Description: SCALE establishes the (floating) ordinate scaling for spectrum plotter routines.

Program Listing:

```
SUBROUTINE SCALE(DMAX, MAXSCL)
C * * * THIS SUBROUTINE ESTABLISHES ORDINATE SCALING FOR
C THE REMOTE SPECTRUM PLOTTER.
   IF(DMAX.LE.0.) GO TO 10
   MAXSCL = 0
   1 MAXSCL = MAXSCL + 10
   DIFF = DMAX - MAXSCL
   IF(DIFF.GT.0.) GO TO 1
   GO TO 999
10 MAXSCL = 0
11 MAXSCL = MAXSCL - 10
   DIFF = (DMINAX - MAXSCL)
   IF(DIFF.LE.0.) GO TO 11
   MAXSCL = MAXSCL + 10
999 RETURN
END
```
27. Subroutine SIGGEN

Called by: PROCES

Calls: none

Commons: blank, CDATA, CDOM

Entries: none

Description: SIGGEN generates and stores in the data array samples of the time function for the specified carrier frequency, modulation frequency, modulation types, and modulation indices. Modulation types include AM, FM, and PM; AM can be used in combination with either of the other two. At least one of the indices for FM and PM must be zero. SIGGEN flags the data array as containing a time function.

Program Listing:
28. Subroutine STRDTA

Called by: MAIN
Calls: none
Commons: CDATA, CFREQ, CFLGS, CCIRKT, CWORD

Description: STRDTA stores the input parameters of all circuit blocks in permanent storage. The incoming data is removed from array WORD (placed there by FETCH) and stored in the basic data storage array DATA. Storage is dynamic, with the data location specified by JCTR. The type of block and the associated value of JCTR is stored external to this routine in the two column array ITYP.

Program Listing:

```fortran
SUBROUTINE STRDTA(K1,K2,K3)
COMMON/CDATA/ JCTR,DATA(200)
COMMON/CFREQ/ NFR,FR(6)
COMMON /CFLGS/ PDFLG,ARFLG
COMMON /CCIRKT/ NBLK,ITYP(30,2)
COMMON /CWORD/ WORD(IC)
LOGICAL PDFLG
JCTR = ITYP(NBLK+1,2)
DO 10 I = 1,K1
  DATA(JCTR+I-1) = WORD(I+1)
  JCTR = JCTR + K1
  IF(K2 .EQ. 0) GO TO 999
  PDFLG = .TRUE.
  DO 20 I = 1,K2
    TEMP = WORD(K3+I)
    IF (TEMP .LT. I) GO TO 20
    NFR = NFR + I
    IF(NFR .GT. 6) GO TO 50
    FR(NFR) = TEMP
  20 CONTINUE
  GO TO 999
  50 WRITE(6,7000)
  STOP
  7000 FORMAT(/- INPUT FREQUENCIES EXCEED SIX/-)
999 RETURN
END
```
29. Subroutine SYNBP

Called by: PROCES
Calls: FRQFCN
Commons: blank, CDATA
Entries: SYNLNP, SYNHP

Description: SYNBP provides a model of a synchronously tuned bandpass filter. Entries SYNLNP and SYNHP provide low pass and high pass models. The filter transfer function is computed to be

\[ H(f) = \left( \frac{1}{1 + jf_p} \right)^n \]

where

\[ n = \text{number of sections}, \]

\[ f_p = \begin{cases} 
\frac{f/f_o'}{co} & \text{for low pass}, \\
- \frac{f'/f}{co} & \text{for high pass}, \\
\left(\frac{f - f_o}{f'_o}\right)/f'_o & \text{for bandpass}, 
\end{cases} \]

\[ f'_o = f_{co}/\hat{\delta}(n), \]

\[ \hat{\delta}(n) = \sqrt{2^{1/n} - 1}, \]

and \( f_{co} \) is the nominal corner frequency in Hz. For this idealized filter, the insertion loss at the center of the pass band is zero, and the attenuation at the corner frequency is 3 dB. The computed transfer function for the bandpass filter is symmetrical about center frequency.
Program Listing:

```fortran
SUBROUTINE SYN6P(A)
COMPLEX A(1), D, DI
COMMON N, IGAM, DELF, DELT
COMMON /COATA/ JCTR, DATA(200)
LOGICAL FFLG
F0 = DATA(JCTR)
FCOFF = DATA(JCTR+1) / 2
NR = DATA(JCTR+2)
FFLG = .TRUE.
GO TO 20
ENTRY SYNLP(A)
FFLG = .TRUE.
GO TO 10
ENTRY SYNHP(A)
FFLG = .FALSE.
10 F0 = 0.
FCOFF = DATA(JCTR)
NR = DATA(JCTR + 1)
20 TEST = EXP(35. / NR)
CALL FRQFCN(A)
X = EXP(.69314718 / NR)
PSI = SQRT(X - 1.)
FCPR = FCOFF / PSI
DO 50 I = 1, N
II = I - 1 - N/2
F = II * D*L TF
FP = SIGN(1.0, F) * (ABS(F) - F0) / FCPR
IF (FFLG) GO TO 30
IF (ABS(FP) .LT. 1.E-18) GO TO 40
FP = -1.0
30 IF (ABS(FP) .GT. TEST) GO TO 40
D1 = CMPLX(1.0, FP)
D = DI * NR
A(I) = A(I) / D
GO TO 50
40 A(I) = CMPLX(0.0, 0.0)
50 CONTINUE
RETURN
END
```
30. Subroutine TELPLT

Called by: MAIN
Calls: TIMFCN
Commons: blank
Entries: none

Description: TELPLT is a routine which provides a remote teletype plot of the time waveform data. The routine provides the capability of plotting selected portions of the array containing the data. This portion of the data array to be plotted is controlled by specification of the input parameters NST, NSP, NJUMP which specify the starting point in the data array, the stopping point, and the number of data array points skipped between plotted points.
Program Listing:

```fortran
SUBROUTINE TELPLT(A,NST,NSP,NJUMP)
COMPLEX A(1)
DIMENSION IA(50)
COMMON N,IGAM,DELF,DELT,PO
CALL TIMFCN(A)
BMAX = REAL(A(NST))
BMIN = BMAX
DO 1 I = NST,NSP,NJUMP
  B = REAL(IA(I))
  IF(B .LT. BMIN) BMIN = B
  IF(B .GT. BMAX) BMAX = B
  IF((BMAX - BMIN) .LT. 1.E-30) GO TO 999
  DO 33 I = 1,50
  IA(I) = 1H
100  WRITE(6,9)
4  FORMAT(5X,1H+)
   WRITE(6,3) BMIN,BMAX
3  FORMAT(12X,-AMPLITUDE-,MIN.,E9.4,-,MAX.,E9.4,- VOLTS-)
   WRITE(6,4)
   WRITE(6,6)
6  FORMAT(6X,2H-0,3X,2H+1,3X,2H+2,3X,2H+3,3X,2H+4,3X,2H+5,
    3X,2H+6,3X,2H+7,3X,2H+8,3X,2H+9,2X,3H+1,0)
   WRITE(6,7)
7  FORMAT(1H,8X,1H1,1U(5H-----I))
   DO 1 I = NST,NSP,NJUMP
   B = REAL(A(I))
   M = INT(B*50.*+.5)
   IF(M.EQ.0) GO TO 34
   IA(M) = 1H*
   WRITE(6,35) I,IA
   IA(M) = 1H
35  FORMAT(6X,15,3H I,5,JA1)
   GO TO 36
34  WRITE(6,37) I
37  FORMAT(2X,15,3H *)
36  CONTINUE
10  CONTINUE
   WRITE(6,11)
11  FORMAT(6X,1HN)
   GO TO 990
999  WRITE(6,9)
9  FORMAT(1X,12HERROK FINISH)
900  RETURN
END
```

172
31. Subroutine TIMFCN

Called by: AMDEMO, FMDEMO, FRQMUL, IDLMUL, LIM, PHDEMO, TELPLT, WRTF

Calls: LFOLD, FFT

Commons: blank, CDOM

Entries: FRQFCN

Description: TIMFCN checks the type of function stored in the data array (indicated by DOMFLG) and transforms when necessary. A call to TIMFCN will assure that a time function is in the data array, while a call to FRQFCN will assure that a frequency function is in the data array.

Program Listing:

```
SUBROUTINE TIMFCN(A)
COMPLEX A(1)
COMMON N, IJAM
COMMON/CDOM/,DOMFLG
LOGICAL DOMFLG
IF(DOMFLG) GO TO 999
CALL LFOLD(A,N)
CALL FFT(A, IJAM, 1)
DOMFLG = .TRUE.
GO TO 999
ENTRY FRQFCN(A)
IF(.NOT. DOMFLG) GO TO 999
CALL FFT(A, IJAM-1)
CALL LFOLD(A,N)
DOMFLG = .FALSE.
999 RETURN
END
```
32. Subroutine TTFP

Called by: MAIN

Calls: FRQFCN, SCALE

Commons: blank

Entries: none

Description: TTFP is a routine which provides a remote teletype plot of the frequency function. The frequency spectrum is plotted between the limits specified as input parameters. These input parameters are FLO and FHI which specify the low and high frequency limits, in Hz, of the spectrum to be plotted.

Program Listing:

SUBROUTINE TTFP(A,FLO,FHI)
C * * THIS SUBROUTINE PROVIDES A TELTYPE PLOT OF THE FREQUENCY
C SPECTRUM FROM FLO TO FHI AND PRINTS THE CARRIER FREQUENCY
COMPLEX A(1)
COMMON N,IGAM,DELF
DIMENSION IA(50),MM(6)
CALL FRQFCN(A)
C * * * *
NST = (N/2) + INT(FLO/DELF + SIGN(*5,FLO))
NST = NST + 1
NSP = (N/2) + INT(FHI/DELF + SIGN(*5,FHI))
NSP = NSP + 1
TTFP (Continued)

C *** ***

DBMAX = -1.30
IEND = NSP - NST + 1
DO 1 I = 1, IEND
DECTMP = CADS(A(NST + I - 1))
IF(DECTMP .LT. 1E-30) DECTMP = 1E-30
0 = 2U. * ALOG10(DECTMP)
IF(0.GT.0DBMAX) JDBMAX = 0
1 CONTINUE
C *** ***
WRITE(6,2) IEND
2 FORMAT(5X,8HNSIZE = ,I5/)
C *** ***
CALL SCALE(DEMAX, MAXSCL)
C *** ***
C THE ORDNATE WILL VARY FROM (MAXSCL - 5U) DB UP
C TO MAXSCL DB.
DO 33 I = 1, 5
33 IA(I) = 1M
DO 5 I = 1, 6
5 M(I) = MAXSCL - 1U*(6-I)
MAXF = ABS(FLO)
IF(ABS(FHI).GT.MAXF) MAXF = ABS(FHI)
NAMEF = 0
IF(MAXF .GT. 1E3) NAMEF = 3
IF(MAXF .GT. 1E6) NAMEF = 6
IF(MAXF .GT. 1E9) NAMEF = 9
IF(NAMEF .EQ. 0) WRITE (6,2U0)
IF(NAMEF .EQ. 3) WRITE (6,2U3)
IF(NAMEF .EQ. 6) WRITE (6,2U6)
IF(NAMEF .EQ. 9) WRITE (6,2U9)
2U0 FORMAT(2X,14HFREQUENCY (HZ), 9X, 8HECIS10ELS)
2U3 FORMAT(2X,15HFREQUENCY (KHZ), 8X, 8HECIS10ELS)
2U6 FORMAT(2X,15HFREQUENCY (MHZ), 8X, 8HECIS10ELS)
2U9 FORMAT(2X,15HFREQUENCY (GHZ), 8X, 8HECIS10ELS)
WRITE(6,7) (MM(I)*1 = 1M)
7 FORMAT(/7X,14,4(6X,14), 5X, 14)
WRITE(6,8)
8 FORMAT(9X,1H1, 5(1H----))))
TTFP (Continued)

FFACT = 1.
IF(NAMEF.EQ.3) FFACT = 1.E-3
IF(NAMEF.EQ.6) FFACT = 1.E-6
IF(NAMEF.EQ.9) FFACT = 1.E-9
FLO = FLO * FFACT
FHI = FHI * FFACT
DELF = DELF * FFACT
FLOPRT = DELF1*(NST -1 -N/2)
DO 10 I = 1, IEND
DECTMP = CABS(A(NST + I - 1))
IF(DECTMP.LT.1.E-30) DECTMP = 1.E-30
U = 20.* ALOG10(DECTMP)
M = 50. + B - MAXSCL
J = I - 1
XJ = J
FREQ = FLOPRT + XJ * DELF1
IF(M.EQ.0) GO TO 59
IF(M.EQ.1) GO TO 34
DO 1515 II = 1,M
1515 IA(II) = IA(II) + 1
1516 IA(II) = 1
95 FORMAT(F8.3, 2H1, 5X, A1)
GO TO 36
34 WRITE(6*37) FREQ
37 FORMAT(1X, F7.3, 2H1)
GO TO 36
50 WRITE(6*51) FREQ
51 FORMAT(1X, F7.3, 2H1)
1000 FORMAT( )
36 CONTINUE
10 CONTINUE
WRITE(6,11)
11 FORMAT(6X, 4HFREQ)
RETURN
END
33. Subroutine WRTF

Called by: MAIN

Calls: TIMFCN, FRQFCN

Commons: blank

Entries: WRFF

Description: WRTF generates a printed listing of the time function between selected limits. Entry WRFF generates a printed listing of the frequency function between selected limits.

Program Listing:

```
SUBROUTINE WRTF(A,N1,N2)
COMPLEX A(I)
COMMON N,IGAM,DELF
CALL TIMFCN(A)
WRITE (6,704)
DO 10 I = N1,N2
10 WRITE(6,705) I,A(I)
GO TO 995
ENTRY WRFF(A,FLUs,FH1)
CALL FRQFCN(A)
NCTR = N/2 + 1
FLOT = FLO - DELF/16
FHIT = FH1 + DELF/16
WRITE(6,702)
DO 20 I = 1,N
F = (I - NCTR)*DELF
IF (F .LT. FLOT) GO TO 20
IF (F .GT. FHIT) GO TO 999
THETA = 1.37
T = COS(A(I))
IF (T .LT. 1.E-30) 60 19
THETA = 57.29578*ATAN2(IMAG(A(I)),REAL(A(I)))
60 DO = 1.E30
IF (T .LT. 1.E-30) DO = 20*ALOGI10(I)
WRITE(6,701) I,DO,A(I),F,THETA
20 CONTINUE
995 WRITE(6,703)
700 FORMAT(1H,15,2(1PE1.3,2X))
701 FORMAT(1H,15,1PE12.4,3(2X,UPF16.5),2X,F7.2,2X,F6.1)
702 FORMAT(1X,L16,S9X,-REAL,-6X,-IMAG,-6X,-PHASE/-)
703 FORMAT(1X)
704 FORMAT(1X,L16,S9X,-REAL,-8X,-IMAG/-)
RETURN
END
```
APPENDIX D

Listing of SIGMA-5 Version of those FATCAT Routines which were Modified to Adapt them to the SIGMA-5.

NOTE: Many of the Hollerith strings in format statements were delimited with quote marks. The printer used to make the following listing did not have the quote character; a minus sign appears where each quote should have been.
MAIN PROGRAM

COMPLEX A(1024)
COMMON *IGAM, DELF, DELT, PO, CARRFQ
COMMON /CFREQ/ NFR, FR(6)
COMMON /CDOM/ DOMFLG
COMMON /CDATA/ JCTR, DATA(200)
COMMON /CCIRKT/ NBLK, ITYP(30, 2)
COMMON /CWORD/ WORD(12)
COMMON /CFLAGS/ PofilG, ARFLG
LOGICAL PofilG, ARFLG
JCTR = 1
ITYP(1:2) = JCTR
NFR = 0
NBLK = 0
Iblk = 0
PofilG = *.FALSE.*
ARFLG = *.FALSE.*
WRITE(6, 7006)
10 DO 2 I = 1, 12
2 WORD(I) = 4 H
CALL FETC(WORD, L, NBAD)
IF (NBAD *EQ. 0) GO TO 1
CALL ELFINJ(WORD, LTYP)
GO TO 10
3 10, 20, 30, 20, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200
2 210, 220, 230, 240, 250, 260, 270, 280, 290, 300
3 310, 320, 330, 340, 350, 360, 370, 380, 390, 400
4 410, 420, 430, LTYP
10 IF (WORD(4) *EQ. 1 H) GO TO 12
N1 = WORD(3)
N2 = WORD(4)
GO TO 15
12 WRITE(6, 7004)
READ(5, 7005) N1, N2
15 CALL WRTF(A, N1, N2)
GO TO 1
20 IF (WORD(4) *EQ. 1 H) GO TO 22
FRLO = WORD(3)
FRHI = WORD(4)
GO TO 25
22 WRITE(6, 7005)
READ(5, 7005) FRLO, FRHI
25 IF (LTYP *EQ. 4) GO TO 40
CALL WRFF(A, FRLO, FRHI)
GO TO 1
C * * TTY TIME PLOT
30 IF (WORD(4) *EQ. 1 H) GO TO 32
NST = WORD(3)
NSP = WORD(4)
NJUMP = WORD(5)
IF (NJUMP *EQ. 1 H) NJUMP = 1
GO TO 35

180
MAIN (Continued)

32 WRITE(6,7007)
   READ(5,7000) NST,NSP,NJUMP
35 IF (NJUMP .LT. 1) NJUMP = 1
   CALL TELPLTIA(NST,NSP,NJUMP)
   GO TO 1
C ** TTY FREQUENCY PLOT
40 CALL TTFP(A,FRLC,FRL1)
   GO TO 1
C ** CALLCMP TIME PLOT
50 WRITE(6,7110) (((ITYP(I*J)* J = 1*2), I = 1*5)
   7101 FORMAT(1X,2(13,3X))
   80 GO TO 650
C ** PRINT PRIME FACTORS
70 CALL PRTFAC
   GO TO 1
C ** END OF JOB
80 GO TO 999
C ** BUTTERWORTH BANDPASS
90 CALL STRDTA(3,0,0)
   NTYP = 3
   GO TO 600
C ** BUTTERWORTH LOWPASS
100 CALL STRDTA(2,0,0)
   NTYP = 4
   GO TO 600
C ** BUTTERWORTH HIGHPASS
110 CALL STRDTA(2,0,0)
   NTYP = 5
   GO TO 600
C ** BUTTERWORTH BANDSTOP
120 CALL STRDTA(3,0,0)
   NTYP = 6
   GO TO 600
C ** CHEBYSHEV BANDPASS
130 CALL STRDTA(4,0,0)
   NTYP = 7
   GO TO 600
C ** CHEBYSHEV LOWPASS
140 CALL STRDTA(3,0,0)
   NTYP = 8
   GO TO 600
C ** CHEBYSHEV HIGHPASS
150 CALL STRDTA(3,0,0)
   NTYP = 9
   GO TO 600
C ** CHEBYSHEV BANDSTOP
160 CALL STRDTA(4,0,0)
   NTYP = 10
   GO TO 600
MAIN (Continued)

C ** SYNCHRONOUS BANDPASS FILTER
170 CALL STRDATA(3,0,0)
       NTYP = 11
       GO TO 600
C ** SYNCHRONOUS LOWPASS FILTER
180 CALL STRDATA(2,0,0)
       NTYP = 12
       GO TO 600
C ** SYNCHRONOUS HIGHPASS FILTER
190 CALL STRDATA(2,0,0)
       NTYP = 13
       GO TO 600
C ** SIGNAL GENERATOR
200 CALL STRDATA(6,2,1)
       CARRFQ = WORD(3)
       NTYP = 1
       GO TO 600
C ** FREQUENCY MULTIPLIER
210 NTYP = 4
       GO TO 600
220 GO TO 430
C ** IDEAL MULTIPLIER
230 CALL STRDATA(2,1,2)
       NTYP = 16
       GO TO 600
240 GO TO 430
C ** FM DEMODULATOR
250 CALL STRDATA(1,0,0)
       NTYP = 19
       GO TO 600
C ** PHASE DEMODULATOR
260 CALL STRDATA(1,0,0)
       NTYP = 20
       GO TO 600
C ** AMPLIFIER
270 CALL STRDATA(1,0,0)
       NTYP = 2
       GO TO 600
C ** LIMITER
280 CALL STRDATA(3,0,0)
       NTYP = 17
       GO TO 600
290 NOUT = WORD(3)
       IF(NOUT LE NBLK) GC TO 291
       WRIT(6,7041) NBLK
       GO TO 1
291 IF(NOUT - 1BLK) 295,295,293
293 IF(PDFL = 0) CALL PDCHA
       ITMP = IBLK + 1
       GO 292 IBLK = ITMP, NOUT
MAIN (Continued)

10TYP = ITYP(1BLK+1)
JCTR = ITYP(1BLK+2)
CALL PROCES(10TYP)
292 CONTINUE
ILOCAL = NOUT
295 WRITE(6,7002) ILOCAL
GO TO 1
300 CONTINUE
310 CONTINUE
320 GO TO 370
330 GO TO 430
C ** AM DEMODULATOR
340 CALL STROTA(1,0,0)
NTYP = 22
GO TO 600
C ** FLAT SPECTRUM GENERATOR
350 CALL STROTA(3,0,0)
CARRFQ = 9.
NTYP = 23
GO TO 600
360 CALL LSTCOM
GO TO 1
370 GO TO 650
380 CALL INPFOR
GO TO 1
390 GO TO 650
400 GO TO 650
410 GO TO 650
420 GO TO 650
430 WRITE(6,7003)
GO TO 1
600 NBLK = NBLK + 1
ITYP(NBLK+1) = NTYP
ITYP(NBLK+1,2) = JCTR
GO TO 1
650 WRITE(6,7100) WORD(1), WORD(2)
GO TO 1
7000 FORMAT(1)
C7000 FORMAT(35x)
7001 FORMAT(= ** ERROR ** LARGEST BLOCK NO IS -12- - * **)
7002 FORMAT(= PROCESSING COMPLETE THRU BLOCK -12-)
7003 FORMAT(= ** UNDEFINED STATEMENT **)
7004 FORMAT(= ENTER LOW, HIGH INDICES-)
7005 FORMAT(= ENTER LOW, HIGH FREQUENCIES-)
7006 FORMAT(= START-)
7007 FORMAT(= ENTER NSTART, NSTOP, NJUMP-)
7100 FORMAT(= COMMAND --A4,A2-- IS NOT YET OPERATIONAL-)
999 CONTINUE
STOP
END
SUBROUTINE FETCH(WORD, LL, NBAD)
INTEGER APOST, BLANK, COMMA, DEcpt, EQUAL, PLUS, RPAREN, TEST
INTEGER BUFFER, BUFFER2, BUFFER3, BCDFPT, TITLE, WORD, E
DIMENSION TITLE(12)
COMMON/CFETCH/, BUFFER2(6), BUFFER1(80)
DIMENSION WORD(1)
EQUIVALENCE (BUFFER3, BUFFER 4)
DATA APOST, BLANK, COMMA, DEpt, EQUAL, MINUS, NINE, NZ, PLUS
   / 1H-, 1H+, 1H, 1H=, 1H-, 1H9, 1H0, 1H+ /
DATA LPAREN, RPAREN
   / 1H(, 1H) /
DATA E / 1HF /
L=0
NCOLS = 80
NBAD = 1

FETCH IS A FREE-FIELD INPUT SUBPROGRAM WHICH RETURNS
THE INPUT DATA IN LL CONSECUTIVE CELLS OF THE
ARRAY WORD. HOLLERITH IS TRUNCATED TO 6 CHAR.

1 CONTINUE
READ (5, 1001, END = 107) ( BUFFER1(I), I=1, 80 )
WRITE (6, 10C5) ( BUFFER1(I), I=1, 50 )

20 M = 0
20 K = 0
N = 0
NCOMA = 2
DO 3 I = 1, 6
3 BUFFER2(I) = BLANK

4 IF ( M-NCOLS ) 40, 100, 100
   * EXAMINE EACH COLUMN, REMOVE BLANKS, AND TEST FOR
SEPARATORS*
   M = M+1
TEST = BUFFER(M)
IF ( TEST = BLANK ) 41, 4, 41
41 IF ( TEST = COMMA ) 42, 6, 42
42 IF ( TEST = EQUAL ) 43, 6, 43
43 IF ( TEST = LPAREN ) 44, 6, 44
44 IF ( TEST = RPAREN ) 45, 4, 45
45 N = N+1
BUFF2(N) = TEST
IF ( K = 0 ) 5, 5, 4
   * IF TYPE HAS NOT BEEN SET (K=0), TEST CHARACTER TO
DETERMINE IF IT IS A DIGIT OR SIGN (NO DECISION),
A 4-B PUNCH (TITLE CARD), A DECIMAL POINT (FLOATING
POINT NUMBER), OR NONE OF THESE. IN WHICH CASE A
HOLLERITH WORD IS ASSUMED. IF K IS SET, IT WILL BE
C WHEN AN INTEGER
1 WHEN A FLOATING POINT NUMBER
2 WHEN A HOLLERITH WORD.
FETCH (Continued)

5 NCOMMA = 1
   IF ( TEST = NZ ) 52,51,51
51 IF ( TEST = NINE ) 4,4,52
52 IF ( TEST = PLUS ) 53,4,53
53 IF ( TEST = MINUS ) 54,4,54
54 IF ( TEST = APOST ) 55,3,55
55 IF ( TEST = DECEPT ) 56,57,56
56 IF ( TEST = E ) 561,560,561
560 IF ( N-1 ) 561,561,4
561 K = 2
   GO TO 4
57 K = 1
   GO TO 4
C
* SELECT MODE OF CONVERSION, BASED UPON K.
6 IF ( K-1 ) 7,7,9
7 BUFF4 = CDFPT (BUFF2,N)
   IF ( N ) 106,106,8
8 WORD (L+1) = BUFF3
   GO TO 91
9 ENCODE (4,1302,WORD (L+1)) (BUFF2(I),I=1,4)
   ENCODE (2,1302,WORD (L+1)) (BUFF2(I),I=5,6)
   L = L + 1
91 L=L+1
C
* IF NOT FINISHED WITH THE CARD IMAGE, REINITIALIZE
   AND CONTINUE. IF THE NCOLS COLUMN CONTAINED A
   COMMA, PROCESS THE NEXT CARD. OTHERWISE, SET THE
   NUMBER OF WORDS CONVERTED IN LL AND RETURN.
   IF ( N-NCOLS ) 2,12,10
10 IF ( TEST = COMMA ) 11,1,11
11 LL=L
   RETURN
C
* MOVE A TITLE CARD INTO THE TITLE ARRAY.
30 ENCODE (4,1301,TITLE) (BUFF1(I),I=1,48)
   GO TO 1
C
* A CARD IMAGE HAS BEEN PROCESSED. IF THE LAST
   NON-BLANK SYMBOL WAS A COMMA (NCOMMA=0), READ
   THE NEXT CARD. OTHERWISE THERE IS INFORMATION
   IN *BUFF2* TO BE CONVERTED, AFTER WHICH, FETCH
   WILL RETURN TO THE CALLING PROGRAM.
100 IF ( NCOMMA ) 6,1,6
C
* FETCH FOUND CONCOMITANT SEPARATORS OR A NUMBER WITH
   MORE THAN 15 DIGITS AND COULD NOT CONTINUE.
106 WRITE (6,2000) (BUFF2(I),I=1,6)
   NBAD = 2
   RETURN
107 STOP
1001 FORMAT (83A1)
1002 FORMAT (6A1)
10035 FORMAT (1X83A1)
25635 FORMAT (25H) ** FETCH CANNOT DECODE 6A1,4H **/**
END
SUBROUTINE INPFOR
COMMON /CWORD/ WORD(12)
INTEGER WORD
LOGICAL FLG
FLG = .TRUE.
CALL ELFIND(WORD(3), L)
GO TO ( 1, 1, 1, 1, 1, 1, 1, 1, 9, 10,
A 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
D 21, 1, 23, 1, 25, 26, 27, 28, 1, 1,
C 1, 1, 1, 34, 35, 1, 1, 1, 1, 1,
D 1, 1, 500), L
1 WRITE(6,7001) WORD(1), WORD(2)
GO TO 999
20 WRITE(6,7020)
WRITE(6,7104)
IF (FLG) GO TO 999
35 WRITE(6,7035)
WRITE(6,7111)
IF (FLG) GO TO 999
9 WRITE(6,7009)
WRITE(6,7106)
IF (FLG) GO TO 999
10 WRITE(6,7010)
WRITE(6,7102)
IF (FLG) GO TO 999
11 WRITE(6,7011)
WRITE(6,7101)
IF (FLG) GO TO 999
12 WRITE(6,7012)
WRITE(6,7111)
IF (FLG) GO TO 999
13 WRITE(6,7013)
WRITE(6,7101)
WRITE(6,7106)
IF (FLG) GO TO 999
14 WRITE(6,7014)
WRITE(6,7102)
WRITE(6,7103)
IF (FLG) GO TO 999
15 WRITE(6,7015)
WRITE(6,7102)
WRITE(6,7103)
IF (FLG) GO TO 999
16 WRITE(6,7016)
WRITE(6,7101)
WRITE(6,7103)
IF (FLG) GO TO 999
INPFOR (Continued)

17 WRITE(6,7017)
   WRITE(6,71u1)
   IF (FLG) GO TO 999
18 WRITE(6,7018)
   WRITE(6,71u2)
   IF (FLG) GO TO 999
19 WRITE(6,7019)
   WRITE(6,71u2)
   IF (FLG) GO TO 999
25 WRITE(6,7025)
   WRITE(6,71v7)
   IF (FLG) GO TO 999
34 WRITE(6,7034)
   WRITE(6,71v7)
   IF (FLG) GO TO 999
26 WRITE(6,7026)
   WRITE(6,71v7)
   IF (FLG) GO TO 999
21 WRITE(6,7021)
   WRITE(6,71u5)
   IF (FLG) GO TO 999
23 WRITE(6,7023)
   WRITE(6,71u6)
   IF (FLG) GO TO 999
27 WRITE(6,7027)
   IF (FLG) GO TO 999
28 WRITE(6,7028)
   WRITE(6,7110)
   GO TO 999
500 IF (WORD(3) .EQ. 3HALL) GO TO 510
   WRITE(6,7112) WORD(3)
   GO TO 999
510 FLG = .FALSE.
   GO TO 20
700 FORMAT()
7001 FORMAT(1X,A4,A2,- IS NOT AN INPUT COMMAND-)
7009 FORMAT(/- DMONUP, FU, dw, NR-)
7010 FORMAT(/- DWLOWP, FC, NR-)
7011 FORMAT(/- DHHIP, FC, NR-)
7012 FORMAT(/- DMDSTP, FU, dw, NR-)
7013 FORMAT(/- CHMDP, FU, dw, NR, EPSDB-)
7014 FORMAT(/- CHLOWP, FC, NR, EPSDB-)
7015 FORMAT(/- CHHIP, FC, NR, EPSDB-)

187
INPFOR (Continued)

7016 FORMAT(/- CHSTP, FU, BW, NR, EPSDB-)
7017 FORMAT(/- SYNBp, FU, dW, NR-)
7018 FORMAT(/- SYNLp, FC, NR-)
7019 FORMAT(/- SYNHP, FC, NR-)
7020 FORMAT(/- SIGGEN, FU, FMOD, AM, PM, FM, A-)
7021 FORMAT(/- FRQMUL-)
7023 FORMAT(/- IDLMUL, ALO, FLO-)
7025 FORMAT(/- FMDEM0, FU-)
7026 FORMAT(/- PHDEMO, Fu-)
7027 FORMAT(/- AMP, GAIN/- GAIN = VOLTAGE GAIN, DB-
   A  - (6 DB = FACTOR OF 2)-)
7028 FORMAT(/- LIM, CL, CH, GL-)
7034 FORMAT(/- AMDEMO, FO-)
7035 FORMAT(/- FLATSP, AMP, DELF, N-)
7101 FORMAT(- FC = CENTER FREQ, HZ/- BW = BANDWIDTH, HZ/-
   A - NR = NUMBER OF SECTIONS-)
7102 FORMAT(- FC = CORNER FREQ, HZ/- NR = NUMBER OF SECTIONS-)
7103 FORMAT(- EPSDB = CHEBYSHEV RIPPLE FACTOR, DB-)
7104 FORMAT(- FU = CARRIER FREQ, HZ/-
   A - FMOD = MODULATION FREQ, HZ/-
   d - AM = PERCENTAGE AMPLITUDE MODULATION-/ C - PM = PEAK PHASE DEVIATION, RADIANS-/ D - FM = PEAK FREQUENCY DEVIATION, HZ-/ E - A = PEAK AMPLITUDE, VOLTS-)
7105 FORMAT(- (NO PARAMETERS)-)
7106 FORMAT(- ALO = PEAK AMPLITUDE OF LO SIGNAL, VOLTS-/
   A - FLO = FREQ OF LO, HZ-)
7107 FORMAT(- Fu = CENTER FREQ, HZ-)
7110 FORMAT(- CL = LOW CLIPPING LEVEL, VOLTS-/
   A - CH = HIGH CLIPPING LEVEL, VOLTS-/ B - GL = LIMITER GAIN, VOLTS/VOLTS-)
7111 FORMAT(- AMP = AMPLITUDE OF SPECTRAL LINES-/
   A - DELF = FREQ SEPARATION OF LINES, HZ-/ B - N = ARRAY SIZE-)
7112 FORMAT(/- INPFOR CANNOT DECODE -A6/-)
999 WRITE(6,7000)
RETURN
END
SUBROUTINE ELFIND(NAME,L)
DIMENSION NAME(1), MATCH(84)
DATA (MATCH(I), I = 1,84)
A/4HPRIN,2HTT,4HPRIN,2HTF,4HTPLO,2HTT,4HTPLO,2HTF,
B 4HCPLO,2HTT,4HCPLC,2HTF,4HPRIM,2HEF,4HENDO,2MFJ,
C 4HdwoN,2HUP,4HdWLC,2HPF,4HdMLI,2HP,4HdWDS,2HTP,
D 4HCdN,2HUP,4HCdLO,2HWP,4HCdM,2HP,4HCdDS,2HTP,
E 4HSYNB,2HP,4HSYNL,2HP,4HSYNm,2HP,4MSIGG,2HEN,
F 4HFRQM,2HUL,4H,2H,4MIDLM,2MUL,4H,2H,
G 4HFMDE,2HMO,4MPJDE,2HMO,4HAMP,2H,4HLIM,2H,
H 4HLOC,2HK,4HYES,2H,4HNO,2H,4HN,2H,
J 4HCIRC,2HUI,4HINSU,2HTF,4HDELE,2HTE,4HINSF,2MRT,
     4HREPL,2HAC,4HREPE,2HAT/
NMAX = 84
DO 11 I = 1,NMAX,2
   IF (NAME(1) = MATCH(I)) 11,5,11
  5 IF (NAME(2) = MATCH(I + 1)) 11,21,11
11 CONTINUE
   I = NMAX + 1
21 L = (I + 1) / 2
RETURN
END
SUBROUTINE LSTCOM
COMMON /CWORD/ WORD(12)
INTEGER WORD
WRITE(6,7001) 1
I = 1
IF (WORD(3) .EQ. 3HALL) GO TO 20
I = 0
IF (WORD(3) .EQ. 4HSOUR) GO TO 20
IF (WORD(3) .EQ. 4HFILT) GO TO 30
IF (WORD(3) .EQ. 4HDEMO) GO TO 40
IF (WORD(3) .EQ. 4HMISC) GO TO 50
IF (WORD(3) .EQ. 4HCONT) GO TO 60
IF (WORD(3) .EQ. 4HNSIZ) GO TO 70
IF (WORD(3) .EQ. 4HLIST) GO TO 80
WRITE(6,702U) WORD(3)
GO TO 999
20 WRITE(6,7002) I
IF (I .EQ. 0) GO TO 999
30 WRITE(6,7003) I
IF (I .EQ. 0) GO TO 999
40 WRITE(6,7004) I
IF (I .EQ. 0) GO TO 999
50 WRITE(6,7005) I
IF (I .EQ. 0) GO TO 999
60 WRITE(6,7006) I
IF (I .EQ. 0) GO TO 999
70 WRITE(6,7007) I
GO TO 999
80 WRITE(6,7008) I
999 RETURN
7001 FORMAT (/15X,-FATCAT COMMAND SUMMARY-)
7002 FORMAT (/10X,-SOURCES-/-SIGGEN-5X,-SIGNAL GENERATOR-/)
   A - FLATSP-5X,-FLAT SPECTRUN GENERATOR-/)
7003 FORMAT (/10X,-FILTERS-/-BUTTERWORT-/-)
   A - BWBNPD-5X,-BAND PASS-/)
   B - BWLOWP-5X,-LOW PASS-/)
   C - BWHIP-6X,-HIGH PASS-/)
   D - BWSTP-5X,-STOP-/)
   E - TCHEBYSHEFF-/)
   F - CHPNDP-5X,-BAND PASS-/)
   G - CHLOWP-5X,-LOW PASS-/)
   H - CHHIP-6X,-HIGH PASS-/)
   I - CHSTP-5X,-STOP-/)
   J - SYNCHRONOUSLY TUNED-/)
   K - SYNBP-6X,-BAND PASS-/)
   L - SYNL-6X,-LOW PASS-/)
   M - SYNH-6X,-HIGH PASS-/)

190
LSTCOM (Continued)

7004 FORMAT(/,1X,DEMODULATORS--//,
   A - FMDEMO-5X-FM DEMODULATOR-/
   B - AMDEMO-5X-AM DEMODULATOR-/
   C - PHDEMO-5X-PHASE DEMODULATOR-/
7005 FORMAT(/,*1.5X-MISCELLANEOUS--//,
   A - FRMUL-5X-FREQUENCY MULTIPLIER-/
   C - IDLMUL-5X-IDEAL MULTIPLIER-/
   D - AMP-8X-AMPLIFIER-/
   E - LIM-8X-LIMITER-/
7006 FORMAT(/,*1.5X-CONTROL COMMANDS--//,
   A - PRINT-5X-PRINT TIME FUNCTION-/
   T - PRINTF-5X-PRINT FREQUENCY FUNCTION-/
   C - TPLOTT-5X-PRINTER PLOT OF TIME FUNCTION-/
   D - TPLOTF-5X-PRINTER PLOT OF FREQUENCY FUNCTION-/
   E - CPLOTT-5X-REMOTE PLOT OF TIME FUNCTION-/
   F - CPLOTF-5X-REMOTE PLOT OF FREQUENCY FUNCTION-/
   G - BLOCK-6X-PROCESS SIGNAL TO OUTPUT OF BLOCK SPECIFIED-/
   H - PRIMEF-5X-LIST PRIME FACTORS OF ALL SOURCE FREQUENCIES-/
   I - End of block-1X-TERMINATES RUN-/
   J - INPUT FORMATS,BLOCKNAME LIST INPUTS FOR NAMED BLOCK-/
7007 FORMAT(/,4X,-SPECIAL COMMANDS TO SPECIFY ARRAY SIZE--//,
   A - YES-18X-LISTED ARRAY SIZE ACCEPTABLE-/
   B - NO-9X-LISTED ARRAY SIZE NOT ACCEPTABLE-/
   C - N-10X-SET ARRAY SIZE TO SPECIFIER VALUE-/
7008 FORMAT(/- LIST COMMANDS COMMANDS ARE LISTED BY GIVING-,
   A - TWO ALPHANUMERIC- WORD SETS SEPARATED BY A COMMA-,/
   B - THE FIRST IS -1H- LIST COMMANDS-1H- POSSIBLE-,
   C - SECOND WORDS AND THE RESULTING OUTPUTS ARE-/
   D - ALL-12X-LIST ENTIRE COMMAND SET-/
   E - SOURCES-18X-LIST COMMANDS FOR SOURCE BLOCKS-/
   F - FILTERS-8X-LIST COMMANDS FOR FILTERS-/
   G - DEMODULATORS-18X-LIST COMMANDS FOR DEMODULATORS-/
   H - MISC-11X-LIST OTHER BLOCK COMMANDS (AMP,LIM,FTC)-/
   I - CONTROL-8X-LIST CONTROL COMMANDS-/
   J - NSIZE-11X-LIST COMMANDS CONTROLLING ARRAY SIZE-/
   K - LIST COMMANDS-2X-LIST THE ABOVE INFORMATION-/
7020 FORMAT(/- LSTCOM CANNOT DECODE --,A6/) END
SUBROUTINE PDCHK
COMMON N, IGAM, DELT, DELT, PD
COMMON /CWORD/ WORD(12)
COMMON /CFLGS/ PDNFLG, ARFLG
LOGICAL PDNFLG, ARFLG
NFLG = 0
NSET = N
CALL PERIOD
IF(ARFLG) GO TO 5
NSET = N
WRITE(6,7040) PD, DELF
10 WRITE(6,7041) N, IGAM, DELT
CALL FETCH(WORD, L, NHEAD)
CALL ELFIND(WORD, LTYP)
IF(LTYP .EQ. 30) GO TO 300
IF(LTYP .EQ. 31) GO TO 320
IF(LTYP .EQ. 32) GO TO 400
IF(LTYP .EQ. 51) STOP
WRITE(6,7022)
GO TO 20
30 WRITE(6,7003)
GO TO 20
40 N = WORD(3)
NFLG = 0
CALL ADJN
IF(N .GT. NSET) GO TO 10
NFLG = 1
GO TO 300
50 IF(NSET .GT. N) GO TO 990
WRITE(6,7005) N
CALL PRTFAC
WRITE(6,7006)
STOP
900 IF(NFLG .EQ. 0) GO TO 999
WRITE(6,7004) N, NSET
WRITE(6,7003)
GO TO 20
990 N = NSET
999 PDNFLG = .FALSE.
ARFLG = .TRUE.
RETURN
7000 FORMAT(/- PERIOD = -1PE13.3,- SECONDS,-, DELTA-F = -,
A E13.3)
7001 FORMAT(- N = -16-, IGAM = -12-, DELTA-T = -
1 *1PE15.3/- IS THIS SATISFACTORY-)
7002 FORMAT(- INPUT MEANINGLESS * ENTER YES, NO, OR N,VALUE-)
7003 FORMAT(- ENTER N, VALUE-)
7004 FORMAT(- N = -16-, UNACCEPTABLE ** N MUST-
1 - BE -16- TO MEET NYQUIST CRITERION-)
7005 FORMAT(- SOURCE FREQUENCIES REQUIRE THE ARRAY SIZE -
1 - TO BE -19//- PRIME FACTORS APEF/)
7006 FORMAT(/- RUN IS BEING TERMINATED-)
END
SUBROUTINE STRDTA(K1,K2,K3)
COMMON/CDATA/ JCTR,DATA(260)
COMMON/CFREQ/ NFR,FR(6)
COMMON /CFLGS/ PDLG,AKFLG
COMMON /CCIRKT/ NDLK,ITYP(3,2)
COMMON /CWORD/ WORD(12)
LOGICAL PDLG
JCTR = ITYP(NDLK + 1,2)
DO 10 I = 1,K1
10 DATA(JCTR + I -1) = WORD(1 + I)
JCTR = JCTR + K1
IF(K2 *GT* 1) GO TO 999
PDLG = .TRUE.*
DO 20 I = 1,K2
TEMP = WORD(K3 + I + 1)
IF (TEMP *LT* 1) GO TO 20
NFR = NFR + 1
IF(NFR *GT* 6) GO TO 50
FR(NFR) = TEMP
20 CONTINUE
GO TO 999
50 WRITE(6,7000)
STOP
7000 FORMAT(/- INPUT FREQUENCIES EXCEED SIX-/) 999 RETURN
END
LSTCOM (Continued)

7064 FORMAT(1X,DEMODULATORS-/
A = FMDEMO-5X,FM DEMODULATOR-
B = AMDEMO-5X,AM DEMODULATOR-
C = PHDEMO-5X,PHASE DEMODULATOR-
7065 FORMAT(1X,MISCELLANEOUS-/
A = FROMUL-5X,FREQUENCY MULTIPLIER-
B = IDMUL-5X,IDEAL MULTIPLIER-
C = AMP-8X,AMPLIFIER-
7066 FORMAT(1X,CONTROL COMMANDS-/
A = PRINTT-5X,PRINT TIME FUNCTION-
B = PRINTF-5X,PRINT FREQUENCY FUNCTION-
C = TPLOTT-5X,PFINTER PLOT OF TIME FUNCTION-
D = TPLOTF-5X,PRINTER PLOT OF FREQUENCY FUNCTION-
E = CPLLOTT-5X,REMOTE PLOT OF TIME FUNCTION-
F = CPLLOTF-5X,REMOTE PLOT OF FREQUENCY FUNCTION-
G = BLOCK-6X,PROCESS SIGNAL TO OUTPUT OF BLOCK SPECIFIED-
H = PRIMEF-5X,LIST PRIME FACTORS OF ALL SOURCE FREQUENCIES-
I = END OF JCR-11X,TERMINATES RUN-
J = INPUT FORMATS, BLOCK NAME LIST INPUTS FOR NAMED BLOCK-
7067 FORMAT(1X,SPECIAL COMMANDS TO SPECIFY ARRAY SIZE-/
A = YES-8X,LISTED ARRAY SIZE ACCEPTABLE-
B = NO-9X,LISTED ARRAY SIZE NOT ACCEPTABLE-
C = N-10X,SET ARRAY SIZE TO SPECIFIER VALUE-
7068 FORMAT(1X,LIST COMMANDS- COMMANDS ARE LISTED BY GIVING-,
A = TWO ALPHANUMERIC- WORD SETS SEPARATED BY A COMMAS-,
B = THE FIRST IS-11H-LIST COMMAND-11H- POSSIBLE-,
C = SECOND WORDS AND THE RESULTING OUTPUTS ARE-/
D = ALL-12X-LIST ENTIRE COMMAND SET-
E = SOURCES-8X-LIST COMMANDS FOR SOURCE BLOCKS-
F = FILTERS-8X-LIST COMMANDS FOR FILTERS-
G = DEMODULATORS-9X-LIST COMMANDS FOR DEMODULATORS-
H = MISC-11X-LIST OTHER BLOCK COMMANDS (AMP, LII, FTC)-
I = CONTROL-8X-LIST CONTROL COMMANDS-
J = N-10X-LIST COMMANDS CONTROLLING ARRAY SIZE-
K = LIST COMMANDS-12X-LIST THE ABOVE INFORMATION-
7069 FORMAT(1X,LSTCOM CANNOT DECODE-14X)
END