GEORGIA INSTITUTE OF TECHNOLOGY

Engineering Experiment Station

PROJECT INITIATION

Date: 11/30/71

Project Title: Study of Mathematical Modeling of Communications Systems Transponders and Receivers

Project No.: A-1379

Project Director: Mr. J. R. Walsh

Sponsor: NASA, Marshall Space Flight Center

Effective ...... November 19, 1971 .... Estimated to run until: ... November 18, 1972

Type Agreement: Contract No. NAS8-28148 .... Amount: $ 48,964 ....

Reports Required: Monthly Progress Reports
Final Technical Report

Contact Persons: Administrative Matters
Mr. R. J. Whitcomb
ONR Resident Representative
Room 276 - New EES Building
Georgia Institute of Technology
Atlanta, Georgia 30332

Technical Matters
Contracting Officer's Representative
(to be appointed by letter)

Assigned to ...... Electronics (Communications) ...... Division

COPIES TO:

- ☐ Project Director
- ☐ Director
- ☐ Associate Director
- ☐ Assistant Director(s)
- ☐ Division Chiefs
- ☐ Branch Head
- ☐ General Office Services
- ☐ Engineering Design Services
- ☐ Photographic Laboratory
- ☐ Research Security Officer
- ☐ Accounting
- ☐ Purchasing
- ☐ Report Section
- ☑ Library
- ☐ Rich Electronic Computer Center
- ☐ ____________________

GTA

pub
dep.
10-5-73

A-3414 N-P

GEORGIA INSTITUTE OF TECHNOLOGY
Engineering Experiment Station

PROJECT TERMINATION

Date February 6, 1973

PROJECT TITLE: Study of Mathematical Modeling of Communications Systems Transponders and Receivers

PROJECT NO: A-1379

PROJECT DIRECTOR: Mr. J. R. Walsh

SPONSOR: NASA - Marshall Space Flight Center

TERMINATION EFFECTIVE: November 18, 1972 (Contract Expiration)

CHARGES SHOULD CLEAR ACCOUNTING BY: November 30, 1972

Contract Closeout Items Remaining: Final Invoice & Closing Documents
Final Report of Inventions
Government Property Inventory & Certificate

COMMUNICATIONS

COPIES TO:

Project Director
Director
Associate Director
Assistant Directors
Division Chief
Branch Head
Accounting
Engineering Design Services

General Office Services
Photographic Laboratory
Purchasing
Report Section
Library
Security
Rich Electronic Computer Center

ENGINEERING EXPERIMENT STATION 225 North Avenue, Northwest · Atlanta, Georgia 30332

7 January 1972

National Aeronautics & Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

Attn: S&E-ASTR-IA

Subj: Monthly Letter Report No. 1, Covering period 19 November 1971 to 31 December 1971, Project A-1379, Contract NAS8-28148, "Study of Mathematical Modeling of Communications Systems Transponders and Receivers," Control No. DCN 1-2-40-23035(IF).

Gentlemen:

Contract NAS8-28148 was undertaken on 19 November 1971 and is for the purpose of developing two sets of mathematical models suitable for the analysis of communication systems. The first set of mathematical models is to be established for the evaluation of communication system transponders and receivers at a block diagram level. The second set of models is for the evaluation of these transponders and receivers at a circuit detail level. Earlier work on Contract NAS8-20054 considered portions of transmitters on the same basis.

A visit was made to Marshall Space Flight Center on 23 November 1971 during which the course of work on the contract was discussed. It was agreed at that time to extend to transponders that portion of the work started on Contract NAS8-20054 which included both the block and circuit detail approaches to communication transmitter modeling. This previous work was concerned specifically with the analysis, on a block level, of an automatic frequency control system of an FM transmitter and, on a circuit detail level, the circuitry of the Airlock Module Transmitter. Also discussed during the visit was the investigation of existing communication system models and circuit analysis programs which may be useful in the development of the models. The circuit analysis program CIRCUS is presently in use on the contract for circuit detail modeling. It was agreed to investigate these other models for the purpose of determining which is best for a specific task and for the purpose of possibly adapting desirable characteristics to a communication system model.

Applicable computer files originally established for the work on contract NAS8-20054 have been reestablished for the present investigation. The files are cataloged on the Fastrand mass storage media of the Georgia Tech Univac 1108 computer. The original files have been screened and only those files and file elements applicable to the present investigation were reestablished.

A model of the first IF amplifier of the CCS transponder receiver was created and added to the files. This model was successfully analyzed using CIRCUS. The run not only provided information on the characteristics of the amplifier, but also served to validate the new CIRCUS files.

A review of the program listing of certain subroutines in CIRCUS has been started. Particular areas being examined are input, output, and monitoring operations. The main objective of examining these details is to find ways of improving the user-model interface by modifying the diagnostic procedures and messages.

During the next period work will continue on the review of the program details of CIRCUS with the goal of providing better diagnostic capabilities and an improvement in the program user interface. Additional receiver circuits will be analyzed at the circuit detail level and formulation of the details of the block approach to transponder analysis will begin. Also during the period an investigation into the availability of other models applicable to communication systems will be started.

The following constitutes a statement of the funding status as of 31 December 1971:

Expenditures to Date

Actual expenditures 19 November 1971 to 31 December 1971 $ 3420

Unexpended Funds to Date

Unexpended funds 31 December 1971 $ 45544

The total unexpended funds are considered adequate for satisfactory completion of the project requirements.

Respectfully submitted:

Joseph R. Walsh, Jr.
Project Director

Approved:

D. W. Robertson, Head
Communications Branch

GEORGIA INSTITUTE OF TECHNOLOGY
ENGINEERING EXPERIMENT STATION 225 North Avenue, Northwest · Atlanta, Georgia 30332

9 February 1972

National Aeronautics & Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

Attn: S&E-ASTR-IA

Subj: Monthly Letter Report No. 2, Covering period 1 January 1972 to 31 January 1972, Project A-1379, Contract NAS8-28148, "Study of Mathematical Modeling of Communications Systems Transponders and Receivers," Control No. DCN 1-2-40-23035(IF).

Gentlemen:

During this period work has continued on the development of mathematical models for communication system transponders and receivers. The review of program details of CIRCUS aimed at improving its diagnostic capabilities was continued. One of the major improvements being sought was some method of identifying components that were contributing to excessive time requirements. Detecting the need for an excessive amount of computer time to run a circuit would not be too difficult, but reliable methods for identifying the components causing the problem have not yet been established. The actual program review was reduced to a relatively low level while some of the factors contributing to excess time requirements were further studied.

The fast time constants associated with small capacitors discharging through small resistors have previously been recognized as contributing to the time problem. The problem can easily arise in transistor circuits due to the small inter-electrode capacitance of many transistors. The cause of transistor related time problems is not easy to recognize since the capacitances associated with the transistor are not readily apparent from the library data.

As an aid to identifying the cause and analyzing transistor related problems, a digital program was constructed to calculate both C_{be} and C_{bc} from the charge control model parameters for any expected base-emitter or base-collector voltage. Tables of both capacitances as a function of the terminal voltages have been calculated for all transistors in the present library. These tables have provided better insight into this class of problems; they should also be valuable in troubleshooting difficulties with computer analysis in the future.

Work on the modeling of receiver circuits was directed toward modeling a mixer and this effort is still underway. The mixer circuit being modeled contains transformers with center tapped windings, and the best method for modeling these for CIRCUS analysis is not clear. A number of experimental runs with models of simple circuits containing center tapped transformers have shown that the choice of coupling coefficients is critical. The circuit can become unstable for some choices; for some others, satisfactory performance was obtained but the computer time required to run the circuit was excessive. A simple procedure for selecting these coefficients is needed and is being sought.

An inquiry was made of the Computer Software Management and Information Center (COSMIC) at the University of Georgia about the availability of programs pertinent to the present investigation. As a result of this inquiry abstracts of eleven related programs were obtained. These abstracts are presently being reviewed to determine the usefulness of a particular program in the present modeling studies.

During the next period work will continue on the study of other computer aided design programs to determine their usefulness for the modeling investigation and on the establishment of techniques for providing better diagnostic capabilities for models presently in use. The circuit detail modeling of selected receiver circuits will also continue.

The following constitutes a statement of the funding status as of 31 January 1972:

Expenditures to Date

Actual expenditures 19 November 1971 to 31 January 1972	$ 6,880

Unexpended Funds to Date

Unexpended funds 31 January 1972	$ 42,084

The total unexpended funds are considered adequate for satisfactory completion of the project requirements.

Respectfully submitted:

Joseph R. Walsh, Jr.
Project Director

Approved:

D. W. Robertson, Head
Communications Branch

GEORGIA INSTITUTE OF TECHNOLOGY
ENGINEERING EXPERIMENT STATION 225 North Avenue, Northwest · Atlanta, Georgia 30332

10 March 1972

National Aeronautics & Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

Attn: S&E-ASTR-IA

Subj: Monthly Letter Report No. 3, Covering period 1 February 1972 to 29 February 1972, Project A-1379, Contract NAS8-28148, "Study of Mathematical Modeling of Communications Systems Transponders and Receivers," Control No. DCN 1-2-40-23035(IF).

Gentlemen:

During this period work continued on the modeling of receiver circuits on a circuit detail basis. Among the circuits investigated were two mixer circuits. These circuits consisted of a transistor switching mixer and a balanced diode mixer. The switching mixer was analyzed in a CIRCUS run. The resulting time waveform of the mixer output exhibited the carrier signal switched on and off at the local oscillator rate. From this waveform the spectral content of the mixer output can be obtained using fast Fourier transform (FFT) techniques.

Some difficulties were encountered with the diode mixer circuit because of the lack of device parameters for typical high frequency mixer diodes. To establish a capability for obtaining these characteristics, the details of the required parameter measurements were investigated. As a part of this effort, an outline was prepared of the measurements required to provide the device parameters for the charge control model of a diode.

Formulation of the modeling of communication receivers on a block basis has also been considered during this period. Block models are envisioned as providing a quick look at selected arrangements of receiver circuits. Such models can provide an insight into the expected performance of a system. The circuit detail models on the other hand provide a detailed analysis of the circuits response characteristics using actual component values. There exists a need for both types of models to provide a flexible analysis capability.

Many receiver circuits can be considered linear because of their small operating voltages. These would include radio frequency and intermediate frequency amplifiers as well as the amplifier following the final detector. Linear circuit analysis techniques operating in either the time domain or the frequency domain could be applied to these circuits.

For other circuits (such as mixers and limiters) which are nonlinear, analysis must be carried out primarily in the time domain. For example, a mixer may be modeled as a product device which generates the product of the time functions representing the received signal and the local oscillator output. After the time function representing the mixer output is obtained, the signal can be transformed to the frequency domain using FFT techniques. Such a transformation enables use of straightforward methods of applying filter functions and gains to the signal.

Work will continue in these two areas of modeling during the next period. Techniques for interfacing plot routines and FFT routines with the circuit detail models will be studied. Block models of various receiver circuits will begin to be assembled, and capabilities of other models will continue to be investigated.

The following constitutes a statement of the funding status as of 29 February 1972:

Expenditures to Date

Actual Expenditures 19 November 1971 to 29 February 1972 $ 11,439

Unexpended Funds to Date

Unexpended funds 29 February 1972 $ 37,525

The total unexpended funds are considered adequate for satisfactory completion of the project requirements.

Respectfully submitted:

J. R. Walsh, Jr.
Project Director

Approved:

D. W. Robertson, Head
Communications Branch

JRW:irn

GEORGIA INSTITUTE OF TECHNOLOGY
ENGINEERING EXPERIMENT STATION 225 North Avenue, Northwest · Atlanta, Georgia 30332

10 April 1972

National Aeronautics & Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

Attn: S&E-ASTR-IA

Subj: Monthly Letter Report No. 4, Covering period 1 March 1972 to 31 March 1972, Project A-1379, Contract NAS8-28148, "Study of Mathematical Modeling of Communications Systems Transponders and Receivers," Control No. DCN 1-2-40-23035(IF).

Gentlemen:

Primary emphasis during this period of work has been in the area of circuit detail modeling. Interfacing the fast Fourier transform (FFT) program with the circuit analysis program, CIRCUS, has been accomplished for the case of signals of known periodicity. This interfacing permits calculation of the frequency spectra present at selected nodes in a circuit. To use this capability a CIRCUS analysis of a circuit is first run until all transients decay and a steady state response is obtained. At this time the solution time increments are selected such that a number of solution values suitable for input to the FFT routine will exactly span one complete period of the time waveform. Accomplishing this required the use of the save and restart capabilities of CIRCUS, locating the storage locations of the data computed by CIRCUS, and placing these data in a computer file which could later be called to provide the input data to the FFT program. As presently implemented, the time function may be evaluated by CIRCUS over an interval larger than that required as input to the FFT. Also, the interfacing program provides the capability of selecting any desired portion of this data file for input to the FFT.

The frequency spectrum produced at the output of a balanced mixer (one selected from the CCS circuitry) was obtained using the combination of the CIRCUS program, the interfacing program, and the FFT program.

During this period several plot routines have been generated or adapted for use with the time and frequency domain programs. Plot routines for generating displays of frequency spectra on either a teletype terminal or a CALCOMP plotter are now available.

The circuit analysis program, ECAP, was obtained during this period. This program is a combination of three programs which provide for (1) dc analysis, (2) ac analysis, and (3) transient analysis. Of most interest presently is the ac analysis program which provides the capability of obtaining the frequency response of a linear circuit. This program will be very useful for the verification of input data to analysis programs such as CIRCUS. It also enables an automated technique for computing transfer function of linear circuits which can be used with frequency domain analysis. Several modifications need to be provided in ECAP to interface it with the other programs. At present the effort required to make these modifications cannot be estimated.

The ac analysis portion of ECAP was tested by evaluating the transfer function of the CCS telemetry bandpass filter. This filter was used since it represents a fairly complex circuit and its transfer function is well known from previous studies. The transfer function obtained with ECAP compared favorably, both theoretically and experimentally, with those previously obtained.

During the next period the major portion of the project effort will be devoted to the block approach to system modeling. This modeling technique will provide a quick look at a proposed receiver or transponder for initial evaluation. Details of the interfacing of the circuit analysis programs will continue to be studied.

The following constitutes a statement of the funding status as of 31 March 1972:

Expenditures to Date

Actual expenditures 19 November 1971 to 31 March 1972 $ 14,699

Unexpended Funds to Date

Unexpended funds 31 March 1972 $ 34,265

The total unexpended funds are considered adequate for satisfactory completion of the project requirements.

Respectfully submitted:

Joseph R. Walsh, Jr.
Project Director

Approved:

D. W. Robertson, Chief
Communications Division

A - 1379

ENGINEERING EXPERIMENT STATION 225 North Avenue, Northwest · Atlanta, Georgia 30332

11 May 1972

National Aeronautics & Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

Attn: S&E-ASTR-IA

Subj: Monthly Letter Report No. 5, Covering period 1 April to 30 April 1972, Project A-1379, Contract NAS8-28148, "Study of Mathematical Modeling of Communications Systems Transponders and Receivers," Control No. DCN 1-2-40-23035(IF).

Gentlemen:

During this period the major portion of the project effort has been directed toward modeling of transponders and receivers on a block basis. Modeling on a block basis differs from the detailed element-by-element simulation (such as that obtained using the analysis program CIRCUS) in that each stage of a communication system is assigned a simplified model or transfer function. When the blocks are linear the powerful techniques available for linear system analysis can be used. The fast Fourier transform (FFT) is one of these techniques and provides the mechanism for switching from the time to frequency domain. Nonlinear blocks are handled by simulation in the time domain.

The block model of a communication system is presently envisioned as the assembly of a standard set of model blocks in any desired order and with a specified set of parameters applicable to each block. To start the construction of such a model the initial effort has been directed toward the construction of a model which consisted of a set of blocks called in sequence. This initial model does not presently allow for parallel signal paths or feedback paths. The assembly of the desired system model is accomplished by a program which reads the type of block desired between specifie modes and then stores the parameters associated with the specific block for future reference during the calculation of the system response. The model operates in both the time and frequency domains. Automatic domain switching, if necessary, is accomplished in the model for each block. A routine which calculates the common period of up to six frequencies is provided for interfacing with the FFT. Establishing this common period assures that all frequency components in a transform will fall on frequencies calculated by the transform and not between these frequencies.

Several model blocks have been constructed and have received varying degrees of checkout during this period. Among these are (1) a signal generator which provides for the generation of AM, PM, or FM signals, (2) a filter which consists of "n" isolated synchronously tuned resonant circuits, (3) an ideal multiplier which forms the product of two signals, (4) an amplifier which simply multiplies the signal of interest by a fixed gain, (5) a frequency multiplier which passes the input signal waveform above a fixed level, and (6) a low-pass filter.

Early in the next period the block model program will be used to assemble a simple communication system test configuration. The model will provide the capability of obtaining output at user specified points in the system being modeled. Either time or frequency responses will be available by a simple call for the output desired at a given point in the system.

During the next period, work will continue largely in the area of block modeling. Additional blocks will be generated and checked out. A visit to MSFC is planned early in the next period during which work on the program will be reviewed.

The following constitutes a statement of the funding status as of 30 April 1972:

<u>Expenditures to Date</u>

Actual expenditures 19 November 1971 to 30 April 1972 $ 19,134

<u>Unexpended Funds to Date</u>

Unexpended funds, 30 April 1972 $ 29,829

The total unexpended funds are considered adequate for satisfactory completion of the project requirements.

Respectfully submitted:

Joseph R. Walsh, Jr.
Project Director

Approved:

D. W. Robertson, Chief
Communications Division

ENGINEERING
GEORGIA INSTITUTE OF TECHNOLOGY
EXPERIMENT STATION 225 North Avenue, Northwest · Atlanta, Georgia 30332

8 June 1972

National Aeronautics & Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

Attn: S&E-ASTR-IA

Subj: Monthly Letter Report #6, Covering period 1 May to 31 May 1972, Project A-1379, Contract NAS8-28148, "Study of Mathematical Modeling of Communications Systems Transponders and Receivers," Control No. DCN 1-2-40-23035(IF).

Gentlemen:

A visit was made to Marshall Space Flight Center on 11 and 12 May 1972, during which work on the contract was reviewed. During the visit progress in the area of communications system modeling and the direction of the project effort for the remainder of the program were the principal topics of discussion. The status of the circuit detail approach, which uses the elaborate computer aided design programs CIRCUS and ECAP, and the block modeling approach, which uses a simplified model for various system functions, was discussed.

During the visit it was agreed that most of the project effort for the remainder of the program should be in the area of block modeling of communication systems. It was also agreed that the effort in the circuit detail modeling should be limited to a summary review of the CIRC-DC, CIRC-AC, and CIRC-TR analysis programs for the Sigma 5 computers, one of which is available at MSFC. Some familiarity with these programs has been obtained during this period by reviewing the program reference manuals.

The majority of the recent project effort has been directed toward the block approach to communication system modeling. A number of model blocks have been developed and several auxiliary subroutines have been generated for block modeling. A control program which allows the calling of the model blocks in any sequential order is also being developed. As an initial test of the model blocks and the control program, a simple amplitude modulated receiver was successfully simulated using the block modeling approach. The program provides the capability of sequential observation of the time waveform or frequency spectrum at any block interface during execution of the program.

In order to develop a more versatile control program, a survey of previously developed digital computer simulation programs for modeling continuous systems has been undertaken. Simulation languages such as DSL/90, CSMP, and MIMIC have been reviewed to determine their applicability to the development of a block-diagram simulator for communication systems.

Preliminary results indicate that each member of this class of programs requires the input of either differential equations or an analog-computer-type detailed block diagram (integrators, summers, etc.) to describe the system being simulated. This is a more complex input procedure than is desired for the block simulator being developed.

Work will continue in these areas during the next period. Review of the Sigma 5 analysis programs will continue at a low level. This review will allow the experience gained with the use of CIRCUS and ECAP to be applied to circuit analysis with the CIRC series programs when they become available. Additional models will be added to those already available for the block models. Among these are additional filter models and models required for the evaluation of frequency and phase modulated receivers.

The following constitutes a statement of the funding status as of 31 May 1972:

Expenditures to Date

Actual expenditures 19 November 1971 to 31 May 1972 .... $ 22,108

Unexpended Funds to Date

Unexpended funds, 31 May 1972 .... $ 26,856

The total unexpended funds are considered adequate for satisfactory completion of the project requirements.

Respectfully submitted:

Joseph R. Walsh, Jr.
Project Director

Approved:

D. W. Robertson, Chief
Communications Division

JRW:irn

ENGINEERING EXPERIMENT STATION 225 North Avenue, Northwest · Atlanta, Georgia 30332

10 July 1972

National Aeronautics & Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

Attn: S&E-ASTR-IA

Subj: Monthly Letter Report No. 7, Covering period 1 June to 30 June 1972, Project A-1379, Contract NAS8-28148, "Study of Mathematical Modeling of Communications Systems Transponders and Receivers," Control No. DCN 1-2-40-23035(IF).

Gentlemen:

During this period the development of block models of communication systems has continued. The major portion of the effort has been directed toward an investigation of techniques for the simulation of those parts of a communication systems containing feedback. Typical examples of these portions are phase-lock loops and automatic gain or frequency control networks. Such feedback systems involve several blocks in the model. The simulation of such devices, which usually contain some nonlinearity, requires that all involved blocks be simulated simultaneously in the time domain. This is in contrast to the block model being developed for cascaded blocks without feedback. The latter model determines the response of each block sequentially and independently in the frequency or time domain, whichever is appropriate.

Simulation languages such as MIMIC are well suited for feedback applications. Using MIMIC as a basis and developing MIMIC subprograms (equivalent to model blocks) for typical communication subsystem blocks such as modulators, filters, mixers, etc., a time domain simulation with simple input/output procedures and efficient (variable time step size) numerical integration will be developed for feedback block diagrams. While the MIMIC system will be applicable to cascaded as well as feedback systems, the block model specifically designed for the simulation of cascaded blocks is more efficient and should be used for non-feedback systems.

The MIMIC language has been adapted to the Univac-1108 computer and techniques have been developed for the use of dynamic subprograms. Such subprograms have been developed for AM modulators, single-tuned filters, mixers, demodulators, and low-pass filters. The subprograms have been debugged and verified by application to a cascaded block diagram system previously simulated by the frequency/time simulator. At present, the use of two block modeling programs, frequency/time for cascaded blocks and MIMIC for feedback systems, seems to be the best approach.

An analysis of stagger-tuned filter design directed toward creating block models for such filters was undertaken. The analysis and modeling of such filters is being pursued in some depth since the quality of the filter models will be a major factor in determining the quality of the overall circuit analysis model. The immediate objective is to create a model block that accurately models a flat staggered n-tuple filter from simple input parameters (center frequency,bandwidth, and number of stages).

Work will continue on the development of block models during the next period. Additional model blocks such as those for various filters should be completed during the coming period. The time domain program for the simulation of feedback systems will be applied to a simulation problem such as a receiver automatic gain control system.

The following constitutes a statement of the funding status as of 30 June 1972:

Expenditures to Date

Actual expenditures 19 November to 30 June 1972	$ 25,735

Unexpended Funds to Date

Unexpended funds, 30 June 1972	$ 23,229

The total unexpended funds are considered adequate for satisfactory completion of the project requirements.

Respectfully submitted:

Joseph R. Walsh, Jr.
Project Director

Approved:

D. W. Robertson, Chief
Communications Division

GEORGIA INSTITUTE OF TECHNOLOGY
Engineering Experiment Station
Systems & Techniques Department
Communications Division

Monthly Progress Report No. 8

1 July to 31 July 1972

by

Joseph R. Walsh, Jr.

Contract No. NAS8-28148
Control No. DCN 1-2-40-23035 (IF)

10 August 1972

ENGINEERING
GEORGIA INSTITUTE OF TECHNOLOGY
EXPERIMENT STATION 225 North Avenue, Northwest · Atlanta, Georgia 30332

10 August 1972

National Aeronautics & Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

Attn: S&E-ASTR-IA

Subject: Monthly Letter Report No. 8, Covering period 1 July to 31 July, 1972, Project A-1379, Contract NAS8-28148, "Study of Mathematical Modeling of Communications Systems Transponders and Receivers," Control No. DCN 1-2-40-23035 (IF).

Gentlemen:

During this period the major portion of the project effort has been devoted to the generation and refinement of model blocks for the communication system block model. This model is being constructed to provide the capability for evaluating the performance of a communication system by the assembly of sequential blocks representing the system functions such as filters, mixers, detectors, etc. These blocks make use of design parameters of the various system functions without having to specify circuit detail parameters. Such parameters as filter center frequency, filter type, and number of sections (or roll-off rate) are the parameters to be specified for a filter instead of the component values which would be specified for a circuit detail model.

Butterworth and Chebyshev bandpass filter blocks have been added to the collection of model blocks during this period. These filter functions are evaluated by first calculating the roots of the denominator of the transfer function, and then evaluating the transfer function at each desired frequency by generating the reciprocal of the product of these roots. Provision has been made for filters of order of 20 or less to be evaluated. Both of these model blocks have been checked and have performed satisfactorily. These bandpass representations have been used as low-pass filters by specification of a zero center frequency and specification of the proper low-pass bandwidth in the call of the filter. A separate model block will be generated which will automatically adjust the input low-pass data for calculation of the low-pass response using the filter bandpass representation. Computation efficiency has been considered during construction of the filter models. The response of a 10 section filter applied to a 4096 element data array can be calculated in less than one second on a Univac-1108 computer.

As new model blocks are generated, appropriate modifications are made to the main control program and to the subroutine which reads and stores the data specifying the parameters needed for a particular model block.

Work is presently underway on the modeling of a frequency modulated receiver. Most of the necessary model blocks have been constructed and checked out. Present emphasis is on the frequency discriminator block. Work will continue on the block model of communication systems during the next period and additional model blocks will be generated. Also the control program will be refined to provide a more useable interface with the user for the conversational mode of operation.

The following constitutes a statement of the funding status as of 31 July 1972:

Expenditures to Date

Actual expenditures 19 November 1971 to 31 July 1972	$29,580

Unexpended Funds to Date

Unexpended funds, 31 July 1972	$19,363

The total unexpended funds are considered adequate for satisfactory completion of the project requirements.

Respectfully submitted:

Joseph R. Walsh, Jr.
Project Director

Approved:

D. W. Robertson, Chief
Communications Division

jrw/jfb

GEORGIA INSTITUTE OF TECHNOLOGY
ENGINEERING EXPERIMENT STATION 225 North Avenue, Northwest · Atlanta, Georgia 30332

11 September 1972

National Aeronautics & Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

ATTN: S&E-ASTR-IA

Subject: Monthly Letter Report No. 9, Covering period 1 August to 31 August, 1972, Project A-1379, Contract NAS8-28148, "Study of Mathematical Modeling of Communications Systems Transponders and Receivers," Control No. DCN 1-2-40-23035 (IF).

Gentlemen:

A visit was made to Marshall Space Flight Center on 17-18 August 1972 during which work on the contract was reviewed. The major topic of discussion was the communication system block model which is being created. This model provides the capability for evaluating certain aspects of communication systems such as the effects of bandwidth, tuning, voltage levels, etc. These systems must be describable by a sequential connection of model blocks. Recent additions to the modeling capabilities have been model blocks which allow simulation of angle modulated systems. The addition of a discriminator block early in this period completed the set of blocks necessary to model a frequency modulation receiver. During the visit, calculated and experimental data obtained for a simulated frequency modulation receiver were discussed and preliminary calculations were presented showing the effects, at baseband, of various filter functions (Butterworth and Chebyshev) inserted in the path of the frequency modulated signal. The time domain simulation of systems containing feedback, such as AFC or AGC systems, using the simulation program MIMIC was also reviewed during the visit.

It was agreed during the visit that a card deck of the block model program would be generated and used to execute the model program on the Sigma 5 computer at MSFC. A tentative date for accomplishing this was agreed to be around 1 October 1972. Any differences between operation on the Georgia Tech Univac 1108 computer, using Fortran V, and the MSFC Sigma 5 computer, which uses extended Fortran IV, can hopefully be resolved during this future visit. These differences are believed to be small since reference to the Fortran IV reference manual is made when programming questions arise.

The time domain block diagram simulator recommended for feedback system simulation utilizes the simulation language MIMIC as described in progress report No. 7. The program has previously been applied to the simulation of a typical AM receiver and is being applied to the simulation of a feedback automatic gain control system. The feedback simulation has been hampered by a large time-constant spread and the system is being temporarily time-scaled for efficient simulation development.

Work during the next period will be largely devoted to refining the model blocks which presently exist and to rewriting portions of the main control program to provide more useable conversational capabilities. The control program presently contains coded input data (for instance, entry of the number "2" is a request for printed output). During the next period an effort will be made to change the user input from coded information to alphanumeric data. Such a change will eliminate the need for the code and provide a more useable program-user interface.

The following constitutes a statement of the funding status as of 31 August 1972.

Expenditures to Date

Actual expenditures 19 November 1971 to 31 August 1972	$34,196

Unexpended Funds to Date

Unexpended funds, 31 August 1972	$14,768

The total unexpended funds are considered adequate for satisfactory completion of the project requirements.

Respectfully submitted:

Joseph R. Walsh, Jr.
Project Director

Approved:

for D. W. Robertson, Chief
Communications Division

jrw/mlw

GEORGIA INSTITUTE OF TECHNOLOGY
ENGINEERING EXPERIMENT STATION 225 North Avenue, Northwest · Atlanta, Georgia 30332

10 October 1972

National Aeronautics & Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

Attn: S&E-ASTR-IA

Subj: Monthly Letter Report No. 10, covering period 1 September to 30 September 1972, Project A-1379 Contract NAS8-28148, "Study of Mathematical Modeling of Communications Systems Transponders and Receivers," Control No. DCN 1-2-40-23035(IF).

Gentlemen:

During this period the effort on the project has been directed toward modeling of communications systems on a block basis. Such a model will allow evaluation of certain characteristics of a communication system at a block diagram level so that major system parameters may be determined before circuit details are considered. The technical activity during this period has been divided among three areas associated with the block model. These are (1) refinement and checkout of model blocks, (2) refinement and checkout of the control program, and (3) development of simulation techniques applicable to communication system components containing feedback.

Many of the individual model blocks used in the communication system block model have been refined during this period. Among these are models of filters and demodulators. The filter models have been designed to use one basic routine for the calculation of the filter response for a specified filter type and a calling routine to obtain the specified filter configuration. For example, a basic Butterworth response for a set of input data would be tailored for the specified configuration of either low-pass, high-pass, bandpass, or bandstop, and the filter response would then be calculated by a routine unique to the filter type called. This procedure is followed for Butterworth, Chebyshev, and synchronously tuned filters. These filter types constitute those presently available in the model. Model blocks for AM, PM, and FM demodulators have also been improved during this period. Checkout of the filter and demodulator models to date indicate that the desired performance is being obtained. Additional checks of the model blocks are in progress.

The control program for the block model has been rewritten to provide a more conversational mode of operation and to provide more flexibility of operation. The numeric codes used in the earlier control program have been replaced by alphabetic abbreviations for model blocks and control functions. The program also provides the capability of entering model blocks one at a time so that a decision on the characteristics of a block to be entered can be based on the results of the calculations up to its input.

An investigation into the use of the simulation program MIMIC for the simulation of communication system components containing feedback is continuing. Time scaling has been applied to an AGC system being simulated to avoid excessive computer time caused by the large time constant spread encountered in such systems. Typical values of the parameters in an AGC system have been selected, and simulation runs are being made using these values and the time scaling.

Work will continue on the control program and the checkout of individual model blocks during the next period. Development of techniques for the simulation of feedback systems will also continue. At present the block model program is operational on the Georgia Tech Univac 1108 computer. No difficulty is anticipated in getting the program operational on the MSFC Univac 1108 computer. It presently appears that some assembly language routines will be required to operate the program on the MSFC Sigma 5 computer. Verification of the translation problems and an investigation of the difficulties involved will be undertaken during the next period.

The following constitutes a statement of the funding status as of 30 September 1972:

<u>Expenditures to Date</u>

Actual expenditures 19 November 1971 to 30 September 1972 $ 37,524

<u>Unexpended Funds to Date</u>

Unexpended funds 30 September 1972 $ 11,440

The total unexpended funds are considered adequate for satisafatory completion of project requirements.

Respectfully submitted:

Joseph R. Walsh, Jr.
Project Director

Approved:

D. W. Robertson, Chief
Communications Division

JRW:iln

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

1972

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 Astrionics 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 OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

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 non-linear 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 output 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 display the data in either the time domain or the frequency domain. Display capabilities include printed listings and plotted time waveforms or frequency 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 exaplined. 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, theyby 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

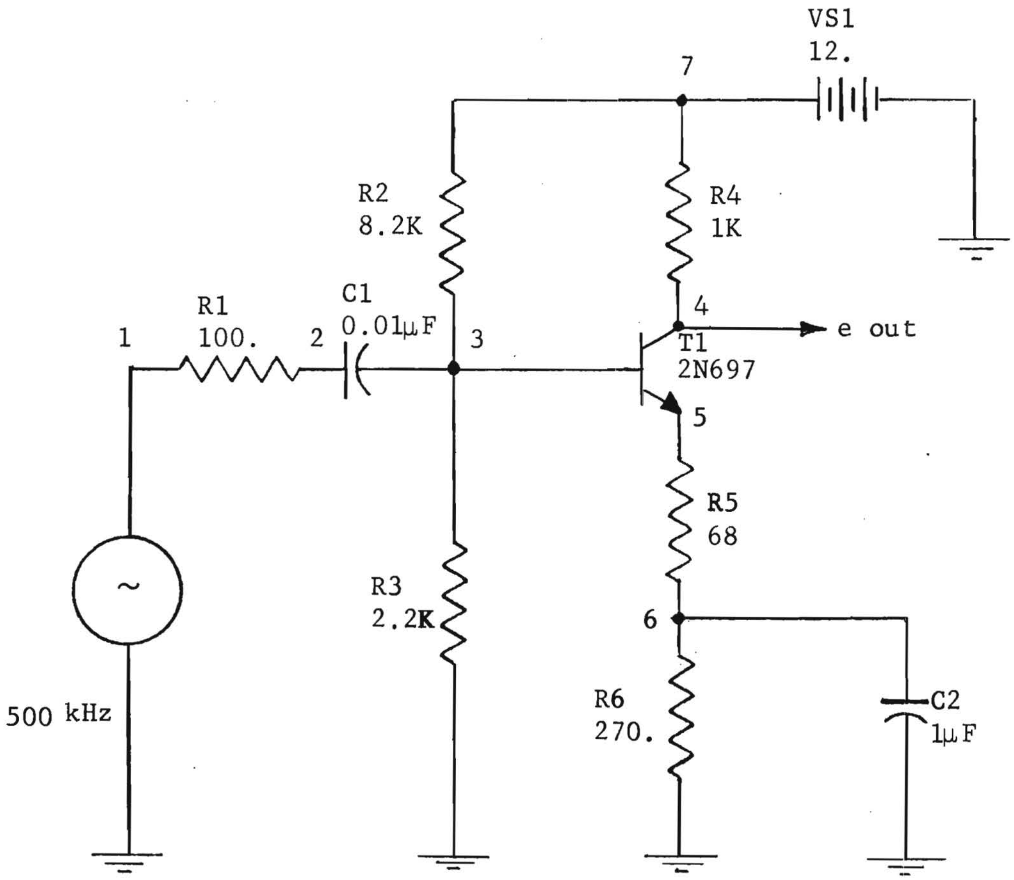


Figure 1. Test Circuit for the CIRCUS Frequency Domain Interface Program.

```
@CAT PFILE.,F2
READY

@ASG,A PFILE.
READY

@USE 19,PFILE
READY

@XQT CAA3.



 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, 0, 6, 270.
 C1, 2, 3, 1.E-8
 C2, 0, 6, 1.E-6
 VS1, 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., RC, 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,
 THETAI, 35.5
   BN, 0., 1.E-5, 1., 68.
   BI, 0., 1.E-5, 1., 4.7
 TCN, .001, 9.93E-10
 TCI, .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.

TIME	VN1	VN2	VN4	IBT1	ICT1
(NSEC)					
.0	0.000	1.676-08	6.91	7.491-05	5.094-03
125.0	.191	.148	5.59	4.245-04	6.407-03
250.0	.354	.296	3.74	4.858-04	8.264-03
375.0	.462	.399	2.38	4.811-04	9.618-03
500.0	.500	.430	2.19	5.375-04	9.806-03
625.0	.462	.414	2.17	3.399-04	9.833-03
750.0	.354	.354	2.15	-1.160-04	9.853-03
875.0	.191	.241	2.84	-5.401-04	9.156-03
(USEC)					
1.000	1.589-08	4.633-02	5.76	-3.996-04	6.241-03
1.125	-.191	-.144	8.46	-2.984-04	3.538-03
1.250	-.354	-.306	10.6	-2.144-04	1.443-03
1.375	-.462	-.420	11.7	-1.022-04	2.884-04
1.500	-.500	-.466	11.9	3.461-06	5.021-05
1.625	-.462	-.436	11.9	6.266-05	9.485-05
1.750	-.354	-.343	11.4	1.624-04	6.221-04
1.875	-.191	-.201	9.95	2.823-04	2.054-03
2.000	-3.179-08	-2.918-02	7.91	3.838-04	4.088-03
2.125	.191	.145	5.72	4.551-04	6.282-03
2.250	.354	.296	3.77	4.886-04	8.228-03
2.375	.462	.400	2.40	4.794-04	9.598-03
2.500	.500	.431	2.19	5.303-04	9.806-03
2.625	.462	.414	2.17	3.332-04	9.832-03
2.750	.354	.355	2.15	-1.222-04	9.852-03
2.875	.191	.240	2.90	-5.305-04	9.102-03
3.000	2.265-07	4.604-02	5.79	-3.955-04	6.206-03
3.125	-.191	-.144	8.49	-2.976-04	3.514-03
3.250	-.354	-.306	10.6	-2.141-04	1.424-03
3.375	-.462	-.419	11.7	-1.013-04	2.773-04
3.500	-.500	-.466	12.0	2.678-06	4.784-05
3.625	-.462	-.435	11.9	6.238-05	8.995-05
3.750	-.354	-.342	11.4	1.616-04	6.091-04
3.875	-.191	-.200	9.96	2.817-04	2.036-03
4.000	-4.212-07	-2.903-02	7.93	3.832-04	4.070-03
4.125	.191	.145	5.74	4.544-04	6.263-03
4.250	.354	.296	3.79	4.876-04	8.211-03
4.375	.462	.400	2.42	4.774-04	9.584-03
4.500	.500	.432	2.19	5.247-04	9.805-03
4.625	.462	.415	2.17	3.278-04	9.832-03
4.750	.354	.356	2.15	-1.271-04	9.852-03
4.875	.191	.239	2.94	-5.232-04	9.061-03
5.000	5.563-07	4.583-02	5.82	-3.925-04	6.178-03
5.125	-.191	-.144	8.50	-2.970-04	3.495-03
5.250	-.354	-.306	10.6	-2.139-04	1.409-03
5.375	-.462	-.419	11.7	-1.007-04	2.683-04
5.500	-.500	-.465	12.0	2.095-06	4.604-05
5.625	-.462	-.435	11.9	6.217-05	8.619-05
5.750	-.354	-.342	11.4	1.609-04	5.989-04
5.875	-.191	-.200	9.98	2.812-04	2.022-03
6.000	-8.106-07	-2.892-02	7.95	3.827-04	4.055-03
6.125	.191	.146	5.75	4.538-04	6.249-03

Figure 2. (Continued).

```
END L5        43

                                   'SINGLE STAGE TEST CIRCUIT'
```

TIME (USEC)	VN1	VN2	VN4	IBT1	ICT1
6.250	.354	.296	3.80	4.867-04	8.197-03
6.375	.462	.400	2.43	4.759-04	9.573-03
6.500	.500	.432	2.19	5.202-04	9.805-03
6.625	.462	.415	2.17	3.235-04	9.832-03
6.750	.354	.356	2.15	-1.310-04	9.851-03
6.875	.191	.239	2.97	-5.176-04	9.028-03
7.000	1.005-06	4.567-02	5.84	-3.901-04	6.157-03
7.125	-.191	-.144	8.52	-2.965-04	3.481-03
7.250	-.354	-.306	10.6	-2.137-04	1.397-03
7.375	-.462	-.419	11.7	-1.002-04	2.611-04
7.500	-.500	-.465	12.0	1.661-06	4.468-05
7.625	-.462	-.435	11.9	6.201-05	8.331-05
7.750	-.354	-.342	11.4	1.604-04	5.908-04
7.875	-.191	-.200	9.99	2.808-04	2.011-03
8.000	-1.319-06	-2.882-02	7.96	3.823-04	4.043-03

```
END OF JOB
 END OF JOB
 ENDJOB        43
END      8049 MLSEC
```

Figure 2. (Continued).

```
@XQT PLT.
 THE NUMBER OF POINTS PER DATA SET =      65
 THE NUMBER OF POINTS USED IN A TRANSFORM =      32
 THE NUMBER OF DATA SETS =   5

 ENTER Ø FOR PRINT OF DATA SET FROM CIRCUS
 ENTER PLUS DATA SET NUMBER FOR FREQ FCN
 ENTER MINUS DATA SET NUMBER FOR TIME FCN
Ø
 ENTER Ø FOR TIME LISTING OR DATA SET NUMBER
Ø

                                   TIME

   Ø.ØØØØ       1.25ØØ-Ø7    2.5ØØØ-Ø7    3.75ØØ-Ø7    5.ØØØØ-Ø7    6.25ØØ-Ø7
   7.5ØØØ-Ø7    8.75ØØ-Ø7    1.ØØØØ-Ø6    1.125Ø-Ø6    1.25ØØ-Ø6    1.375Ø-Ø6
   1.5ØØØ-Ø6    1.625Ø-Ø6    1.75ØØ-Ø6    1.875Ø-Ø6    2.ØØØØ-Ø6    2.125Ø-Ø6
   2.25ØØ-Ø6    2.375Ø-Ø6    2.5ØØØ-Ø6    2.625Ø-Ø6    2.75ØØ-Ø6    2.875Ø-Ø6
   3.ØØØØ-Ø6    3.125Ø-Ø6    3.25ØØ-Ø6    3.375Ø-Ø6    3.5ØØØ-Ø6    3.625Ø-Ø6
   3.75ØØ-Ø6    3.875Ø-Ø6    4.ØØØØ-Ø6    4.125Ø-Ø6    4.25ØØ-Ø6    4.375Ø-Ø6
   4.5ØØØ-Ø6    4.625Ø-Ø6    4.75ØØ-Ø6    4.875Ø-Ø6    5.ØØØØ-Ø6    5.125Ø-Ø6
   5.25ØØ-Ø6    5.375Ø-Ø6    5.5ØØØ-Ø6    5.625Ø-Ø6    5.75ØØ-Ø6    5.875Ø-Ø6
   6.ØØØØ-Ø6    6.125Ø-Ø6    6.25ØØ-Ø6    6.375Ø-Ø6    6.5ØØØ-Ø6    6.625Ø-Ø6
   6.75ØØ-Ø6    6.875Ø-Ø6    7.ØØØØ-Ø6    7.125Ø-Ø6    7.25ØØ-Ø6    7.375Ø-Ø6
   7.5ØØØ-Ø6    7.625Ø-Ø6    7.75ØØ-Ø6    7.875Ø-Ø6    8.ØØØØ-Ø6

 ENTER Ø FOR PRINT OF DATA SET FROM CIRCUS
 ENTER PLUS DATA SET NUMBER FOR FREQ FCN
 ENTER MINUS DATA SET NUMBER FOR TIME FCN
Ø
 ENTER Ø FOR TIME LISTING OR DATA SET NUMBER
3

                              DATA SET NUMBER    3

   6.9Ø6Ø+ØØ    5.5934+ØØ    3.7362+ØØ    2.3816+ØØ    2.1936+ØØ    2.1672+ØØ
   2.1467+ØØ    2.8444+ØØ    5.7588+ØØ    8.462Ø+ØØ    1.Ø557+Ø1    1.1712+Ø1
   1.195Ø+Ø1    1.19Ø5+Ø1    1.1378+Ø1    9.9465+ØØ    7.9117+ØØ    5.7183+ØØ
   3.7717+ØØ    2.4Ø17+ØØ    2.1941+ØØ    2.1675+ØØ    2.1475+ØØ    2.898Ø+ØØ
   5.7943+ØØ    8.4859+ØØ    1.Ø576+Ø1    1.1723+Ø1    1.1952+Ø1    1.191Ø+Ø1
   1.1391+Ø1    9.9642+ØØ    7.93Ø4+ØØ    5.7368+ØØ    3.789Ø+ØØ    2.4159+ØØ
   2.1945+ØØ    2.1677+ØØ    2.1481+ØØ    2.9394+ØØ    5.8219+ØØ    8.5Ø47+ØØ
   1.Ø591+Ø1    1.1732+Ø1    1.1954+Ø1    1.1914+Ø1    1.14Ø1+Ø1    9.9782+ØØ
   7.9452+ØØ    5.7514+ØØ    3.8Ø28+ØØ    2.4272+ØØ    2.1948+ØØ    2.1679+ØØ
   2.1486+ØØ    2.9717+ØØ    5.8435+ØØ    8.5195+ØØ    1.Ø6Ø3+Ø1    1.1739+Ø1
   1.1955+Ø1    1.1917+Ø1    1.14Ø9+Ø1    9.9891+ØØ    7.9569+ØØ

 ENTER Ø 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 3. Execution of Interface Program.

```
ENTER ISTART
32
THE PERIOD OF THE TIME FUNCTION =    4.0000-06 SEC

ENTER 100 FOR PRINT, 010 FOR CALCOMP PLOT, OR 001 FOR TTY PLOT
100
```

LINE	REAL	IMAG	DB
1	-2.7697-02	0.0000	-31.15
2	-4.9144-04	6.6766-04	-61.63
3	2.0091-02	7.9110-03	-33.31
4	-9.1365-05	-6.3436-04	-63.86
5	-2.3060-02	9.5255-03	-32.06
6	-6.6337-04	2.3214-04	-63.06
7	2.9155-02	-4.5856-02	-25.30
8	-1.9304-04	-1.1629-03	-58.57
9	-7.8478-02	-1.6430-03	-22.10
10	-5.3355-04	3.9614-04	-63.55
11	2.3091-01	1.2146-02	-12.72
12	1.0943-03	-1.6229-03	-54.17
13	-1.3070-01	3.1161-02	-17.43
14	-1.7313-03	-8.6959-05	-55.22
15	1.4738+00	-2.2625+00	8.63
16	-8.8963-04	-3.4519-03	-48.96
17	6.9555+00	0.0000	16.85
18	-8.8963-04	3.4519-03	-48.96
19	1.4738+00	2.2625+00	8.63
20	-1.7313-03	8.6959-05	-55.22
21	-1.3070-01	-3.1161-02	-17.43
22	1.0943-03	1.6229-03	-54.17
23	2.3091-01	-1.2146-02	-12.72
24	-5.3355-04	-3.9614-04	-63.55
25	-7.8478-02	1.6430-03	-22.10
26	-1.9304-04	1.1629-03	-58.57
27	2.9155-02	4.5856-02	-25.30
28	-6.6337-04	-2.3214-04	-63.06
29	-2.3060-02	-9.5255-03	-32.06
30	-9.1365-05	6.3436-04	-63.86
31	2.0091-02	-7.9110-03	-33.31
32	-4.9144-04	-6.6766-04	-61.63

```
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 3. (Continued).

```
ENTER ISTART
32
ENTER 100 FOR PRINT, 010 FOR CALCOMP PLOT, OR 001 FOR TTY PLOT
100
```

LINE	REAL	IMAG	DB
1	9.9642+00	0.0000	19.97
2	7.9304+00	0.0000	17.99
3	5.7368+00	0.0000	15.17
4	3.7890+00	0.0000	11.57
5	2.4159+00	0.0000	7.66
6	2.1945+00	0.0000	6.83
7	2.1677+00	0.0000	6.72
8	2.1481+00	0.0000	6.64
9	2.9394+00	0.0000	9.37
10	5.8219+00	0.0000	15.30
11	8.5047+00	0.0000	18.59
12	1.0591+01	0.0000	20.50
13	1.1732+01	0.0000	21.39
14	1.1954+01	0.0000	21.55
15	1.1914+01	0.0000	21.52
16	1.1401+01	0.0000	21.14
17	9.9782+00	0.0000	19.98
18	7.9452+00	0.0000	18.00
19	5.7514+00	0.0000	15.20
20	3.8028+00	0.0000	11.60
21	2.4272+00	0.0000	7.70
22	2.1948+00	0.0000	6.83
23	2.1679+00	0.0000	6.72
24	2.1486+00	0.0000	6.64
25	2.9717+00	0.0000	9.46
26	5.8435+00	0.0000	15.33
27	8.5195+00	0.0000	18.61
28	1.0603+01	0.0000	20.51
29	1.1739+01	0.0000	21.39
30	1.1955+01	0.0000	21.55
31	1.1917+01	0.0000	21.52
32	1.1409+01	0.0000	21.15

```
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 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 questions 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 voltage, from CIRCUS was requested.

The next response was a +3 requesting the frequency domain representation 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 number 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 Calcomp 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 ISTART
32
 THE PERIOD OF  THE TIME FUNCTION =     4.0000-06 SEC

 ENTER 100 FOR PRINT, 010 FOR CALCOMP PLOT, OR 001 FOR TTY PLOT
001
 ENTER FLO
-3.75E6
 ENTER FHI
3.75E6

       NSIZE =      31

  FREQUENCY (MHZ)           DECIBELS

           -30       -20       -10         0        10        20
            I----+----I----+----I----+----I----+----I----+----I
  -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------------
   -.750 I
   -.500 I--------------------------------------
   -.250 I
    .000 I----------------------------------------------
    .250 I
    .500 I--------------------------------------
    .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.

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

```
 ENTER ISTART
32
 ENTER 100 FOR PRINT, 010 FOR CALCOMP PLOT, OR 001 FOR TTY PLOT
001
 ARRAY SIZE =     32  NSTOP MUST BE EQUAL TO OR LESS THAN THIS VALUE

 ENTER NSTART
1
 ENTER NSTOP
32
 ENTER NJUMP
1
  +
         AMPLITUDE: MIN  .0000   , MAX  .1196+02 VOLTS
  +
      0   .1   .2   .3   .4   .5   .6   .7   .8   .9  1.0
      I----I----I----I----I----I----I----I----I----I----I
   1  I                                         *
   2  I                                *
   3  I                       *
   4  I               *
   5  I         *
   6  I        *
   7  I        *
   8  I        *
   9  I           *
  10  I                       *
  11  I                                   *
  12  I                                           *
  13  I                                                *
  14  I                                                 *
  15  I                                                 *
  16  I                                               *
  17  I                                         *
  18  I                                *
  19  I                       *
  20  I               *
  21  I         *
  22  I        *
  23  I        *
  24  I        *
  25  I           *
  26  I                       *
  27  I                                   *
  28  I                                           *
  29  I                                                *
  30  I                                                 *
  31  I                                                 *
  32  I                                               *
   N
 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 5. Time Waveform of Data Set 3.

```
 ENTER ISTART
32
 THE PERIOD OF THE TIME FUNCTION =    4.0000-06 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 0 FOR PRINT OF DATA SET FROM CIRCUS
 ENTER PLUS DATA SET NUMBER FOR FREQ FCN
 ENTER MINUS DATA SET NUMBER FOR TIME FCN
@EOF

PLOT      2.7 MIN      1.3 FT    00F    112072191104
END      1348 MLSEC
```

Figure 6. Request for Calcomp Plot.

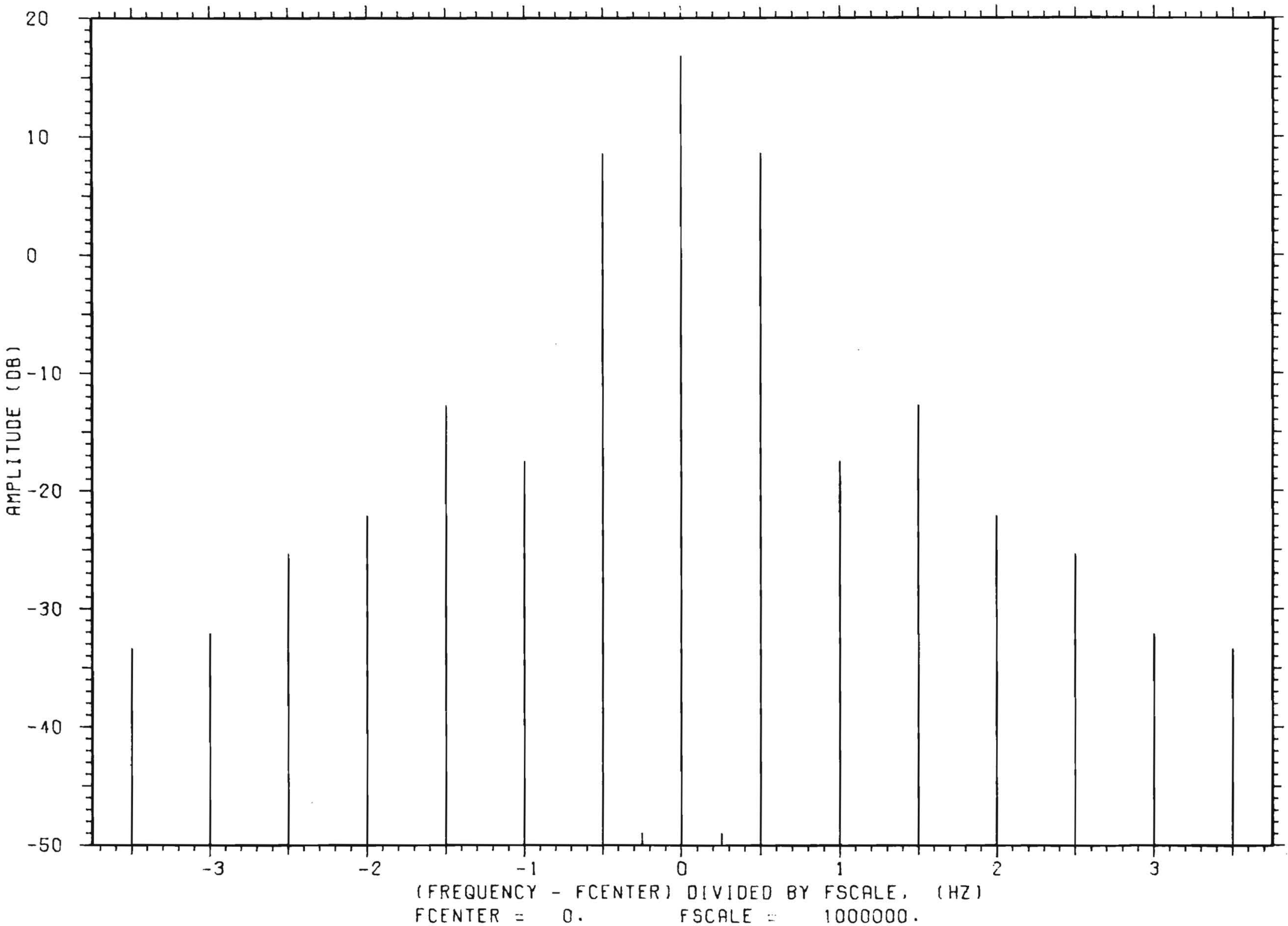


Figure 7. Calcomp Plot of the Frequency Spectrum of the Time Waveform Shown in Figure 5.

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 CCS 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

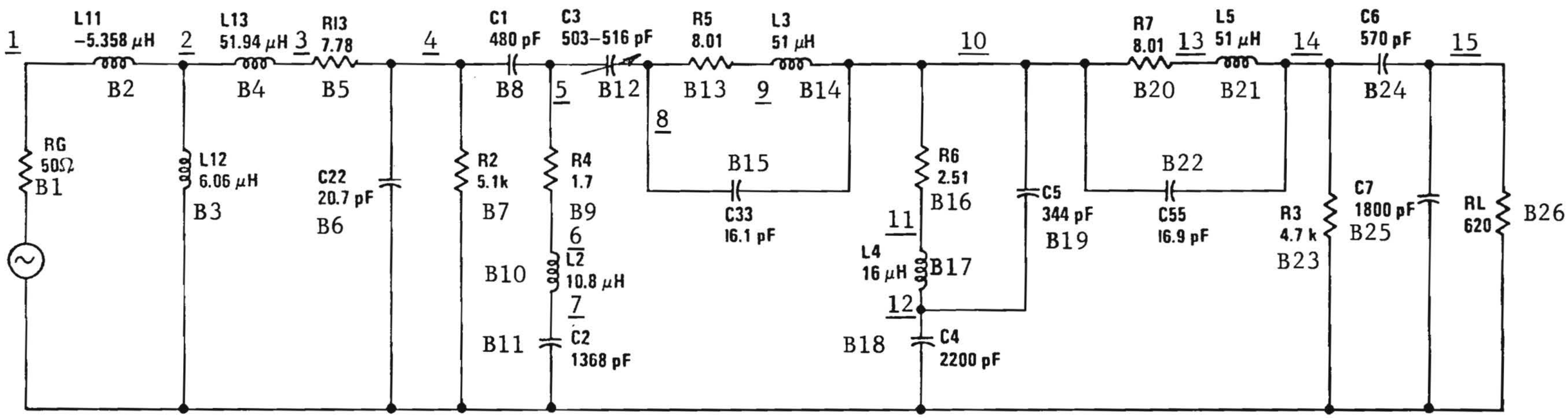


Figure 8. Circuit Diagram of CCS Telemetry Filter.

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.

```
C       CCS TELEMETRY BANDPASS FILTER TRANSFER FUNCTION
C
        AC ANALYSIS
C
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 = 7.78
B6      N(0,4), C = 20.7E-12
B7      N(0,4), R = 5.1E3
B8      N(4,5), C = 480.E-12
B9      N(5,6), R = 1.7
B10     N(6,7), L = 10.8E-6
B11     N(0,7), C = 1368.E-12
B12     N(5,8), C = 510.E-12
B13     N(8,9), R = 8.01
B14     N(9,10), L = 51.E-6
B15     N(8,10), C = 16.1E-12
B16     N(10,11), R = 2.51
B17     N(11,12), L = 16.E-6
B18     N(0,12), C = 2200.E-12
B19     N(10,12), C = 344.E-12
B20     N(10,13), R = 8.01
B21     N(13,14), L = 51.E-6
B22     N(10,14), C = 16.9E-12
B23     N(0,14), R = 4.7E3
B24     N(14,15), C = 570.E-12
B25     N(0,15), C = 1800.E-12
B26     N(0,15), R = 620.
        FREQUENCY = 1024.E3
        PRINT, VOLTAGES
        MODIFY
        FREQUENCY = .5E6(+25)3.E6
        EXECUTE
        END
```

Figure 9. Circuit Description in ECAP Format.

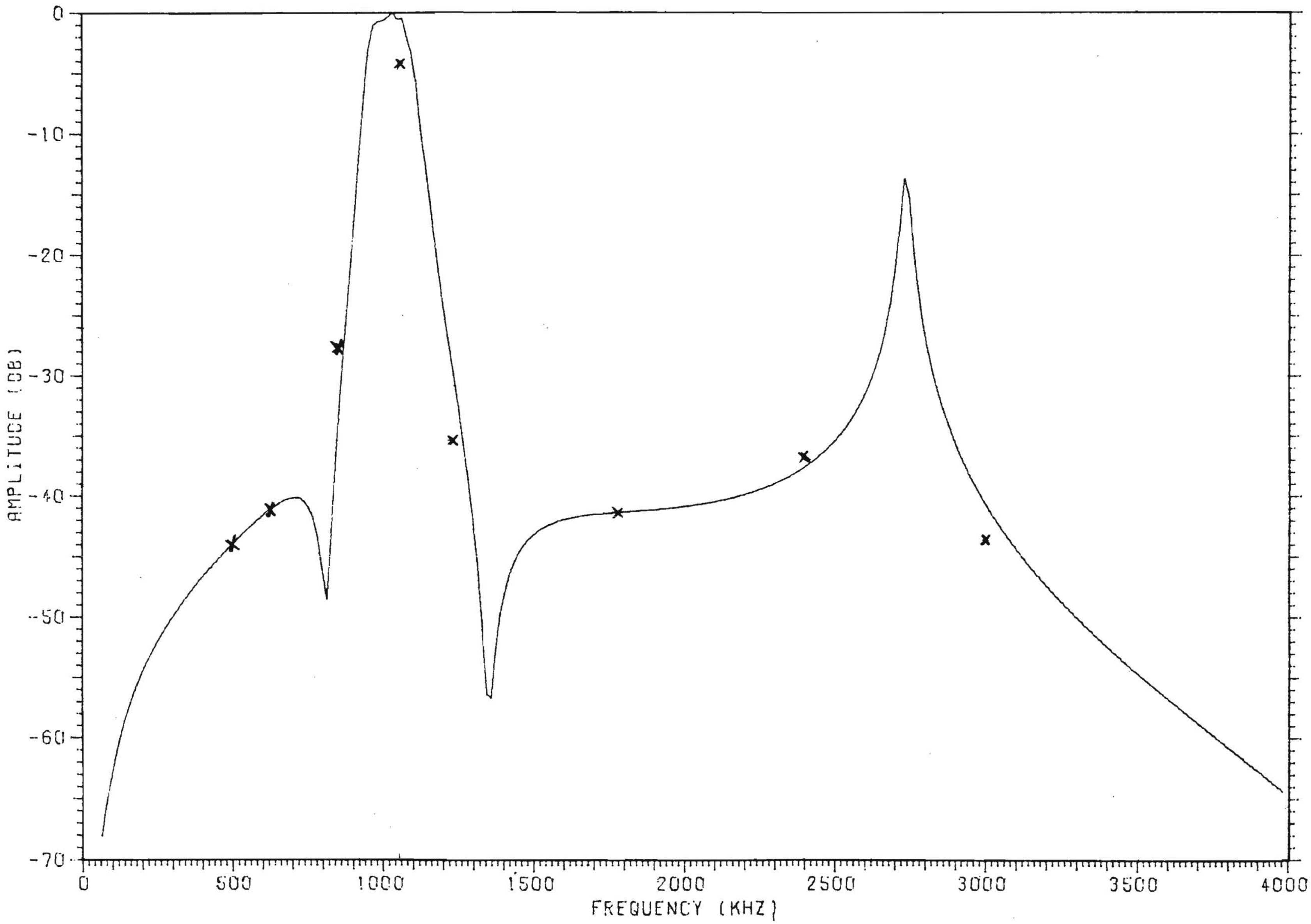


Figure 10. Comparison of Response Calculation for CCS Telemetry.

```
FREQ =     .50000000+06

        NODES            NODE VOLTAGES

MAG    1-  4     .56120345-01    .39131129+00    .49595299+00    .49516120+00
PHA              .84115226+02    .86403404+02    .83893297+02    .83313907+02

MAG    5-  8     .89044465-01    .89041206-01    .10424143+00    .13792944-01
PHA              .83921702+02    .83431645+02    .83431645+02   -.10674115+03

MAG    9- 12     .13643413-01    .12976461-01    .12969306-01    .20500370-01
PHA             -.10122808+03    .86365494+02    .84704815+02    .84609381+02

MAG   13- 15     .12923102-01    .16596556-01    .39010933-02
PHA              .85551819+02    .81962884+02    .94185616+02

FREQ =     .53875705+06

        NODES            NODE VOLTAGES

MAG    1-  4     .63436987-01    .42413890+00    .56286125+00    .56195544+00
PHA              .83260122+02    .85906525+02    .83036861+02    .82408597+02

MAG    5-  8     .97972777-01    .97968375-01    .11793465+00    .22591508-01
PHA              .83089269+02    .82546303+02    .82546303+02   -.10527262+03

MAG    9- 12     .22455424-01    .13817682-01    .13806565-01    .24243406-01
PHA             -.10101118+03    .85583231+02    .83577266+02    .83442150+02

MAG   13- 15     .13754998-01    .18506609-01    .43636454-02
PHA              .84678037+02    .80600175+02    .91967359+02
```

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, demodulators, 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 parameters. 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 domain and the frequency domain. The signal being processed is stored in a complex array as either a frequency domain representation of the signal 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 currently 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 $\beta = 2$, 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

FLATSP, AMP, DELF, N

where

AMP = amplitude of spectrum lines,

DELF = frequency separation of spectral lines (Hz), and

N = array size.

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

BWLOWP, FC, NR

CHLOWP, FC, NR, EPSDB

SYNLP, FC, NR

where

FC = corner frequency (Hz),

NR = number of filter sections, and

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

AMP, Gain

where

Gain = amplifier gain in dB.

A limiter is specified with

LIM, CL, CH, GL

where

CL = low clipping level (volts),

CH = high clipping level (volts), and

GL = limiter gain (volts/volt).

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

FRQMUL

(no parameters necessary).

An ideal multiplier can be specified with

IDLMUL, ALO, FLO

where

ALO = amplitude of the LO signal (volts), and

FLO = 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

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

PRINTT, 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

PRINTT

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

PRINTF, FLO, FHI

where

FLO = low frequency limit (Hz), and

FHI = high frequency limit (Hz).

Alternatively, the command

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

TPLOTT, NSTART, NSTOP, NJUMP

where

NJUMP = interval between plotted points and the other parameters are the same as for PRINTT.

Printer plots of the frequency spectrum are generated by

TPLOTF, FLO, FHI

where FLO and FHI are the same as for PRINTF.

Plotter outputs are similarly generated with

CPLOTT, NSTART, NSTOP

for the time function, and

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 listing 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 instruction 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 command

INPUT FORMAT, BWBNDP

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.

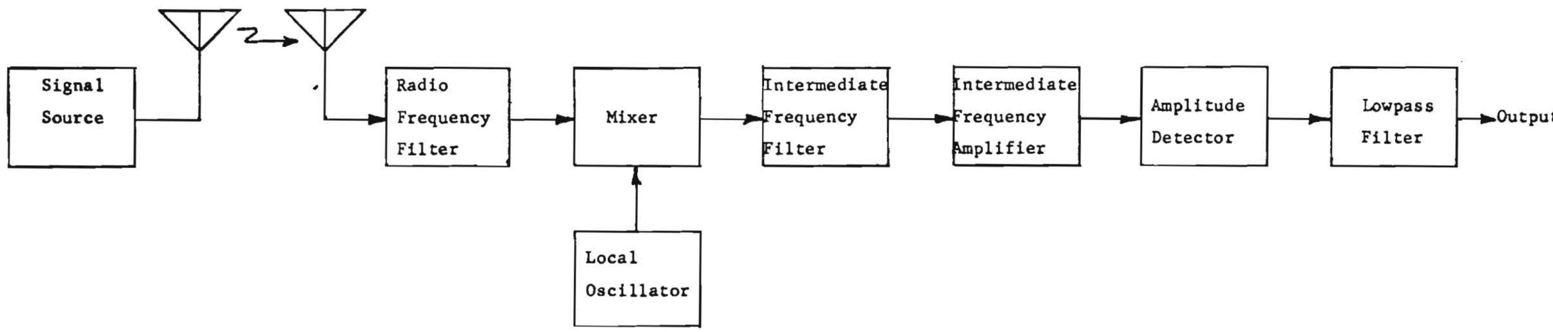
C. Simulation Examples using FATCAT

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.

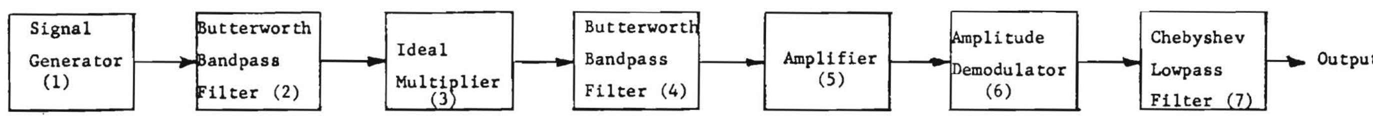
1. AM Receiver

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



(a) Block Diagram of a Typical AM Receiver.



(b) Sequence of Blocks called in FATCAT to Model AM Receiver.

Figure 12. Block Diagram of Amplitude Modulation Receiver and FATCAT Model of Receiver.

```
@XQT P.U
 START
SIGGEN, 1.E7, 1.E4, 50., 0., 0., 1.E-6
 SIGGEN, 1.E7, 1.E4, 50., 0., 0., 1.E-6
BLOCK, 1
 BLOCK, 1

 PERIOD =     1.000-04 SECONDS, DELTA-F =    1.000+04
 N =    2048, IGAM = 11 , DELTA-T =    4.883-08
 IS THIS SATISFACTORY
YES
 YES
 PROCESSING COMPLETE THRU BLOCK  1
TPLOTF
 TPLOTF
 ENTER LOW, HIGH FREQUENCIES
9.9E6, 10.1E6

     NSIZE =     21

  FREQUENCY (MHZ)          DECIBELS

       -170      -160      -150      -140      -130      -120
         I----+----I----+----I----+----I----+----I----+----I
  9.900 I
  9.910 I
  9.920 I
  9.930 I
  9.940 I
  9.950 I
  9.960 I
  9.970 I
  9.980 I
  9.990 I------------------------------
 10.000 I------------------------------------------
 10.010 I------------------------------
 10.020 I
 10.030 I
 10.040 I
 10.050 I
 10.060 I
 10.070 I
 10.080 I
 10.090 I
 10.100 I
    FREQ
```

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
```

LINE	FREQ	REAL	IMAG	MAG	DB	PHASE
2015	9.9000+06	.000	.000	.000	-236.18	87.4
2016	9.9100+06	-.000	.000	.000	-234.92	92.1
2017	9.9200+06	.000	.000	.000	-234.62	84.1
2018	9.9300+06	.000	.000	.000	-233.22	82.6
2019	9.9400+06	-.000	.000	.000	-232.73	96.0
2020	9.9500+06	.000	.000	.000	-230.36	88.2
2021	9.9600+06	.000	.000	.000	-228.22	89.0
2022	9.9700+06	-.000	.000	.000	-227.58	92.0
2023	9.9800+06	.000	.000	.000	-221.96	82.1
2024	9.9900+06	-.000	-.000	.000	-138.06	-90.0
2025	1.0000+07	-.000	-.000	.000	-126.02	-90.0
2026	1.0010+07	-.000	-.000	.000	-138.06	-90.0
2027	1.0020+07	.000	-.000	.000	-222.49	-81.4
2028	1.0030+07	-.000	-.000	.000	-228.82	-92.2
2029	1.0040+07	.000	-.000	.000	-229.39	-88.6
2030	1.0050+07	.000	-.000	.000	-231.95	-88.3
2031	1.0060+07	-.000	-.000	.000	-235.10	-98.4
2032	1.0070+07	.000	-.000	.000	-235.84	-78.3
2033	1.0080+07	.000	-.000	.000	-237.52	-80.1
2034	1.0090+07	-.000	-.000	.000	-238.36	-96.6
2035	1.0100+07	.000	-.000	.000	-239.76	-86.4

Figure 14. Listing of the Frequency Function at the Output of the Signal Generator.

```
BWBNDP, 1.E7, 2.E4, 1
 BWBNDP, 1.E7, 2.E4, 1
BLOCK, 2
 BLOCK, 2
 PROCESSING COMPLETE THRU BLOCK   2
PRINTF
 PRINTF
 ENTER LOW, HIGH FREQUENCIES
9.9E5←6, 10.1E6
```

LINE	FREQ	REAL	IMAG	MAG	DB	PHASE
2015	9.9000+06	-.000	.000	.000	-256.22	171.7
2016	9.9100+06	-.000	.000	.000	-254.06	175.8
2017	9.9200+06	-.000	.000	.000	-252.75	167.0
2018	9.9300+06	-.000	.000	.000	-250.21	164.5
2019	9.9400+06	-.000	.000	.000	-248.41	176.5
2020	9.9500+06	-.000	.000	.000	-244.51	166.9
2021	9.9600+06	-.000	.000	.000	-240.53	164.9
2022	9.9700+06	-.000	.000	.000	-237.58	163.6
2023	9.9800+06	-.000	.000	.000	-228.95	145.6
2024	9.9900+06	.000	-.000	.000	-141.07	-45.0
2025	1.0000+07	-.000	-.000	.000	-126.02	-90.0
2026	1.0010+07	-.000	-.000	.000	-141.07	-135.0
2027	1.0020+07	-.000	-.000	.000	-229.48	-144.9
2028	1.0030+07	-.000	-.000	.000	-238.82	-163.7
2029	1.0040+07	-.000	-.000	.000	-241.69	-164.6
2030	1.0050+07	-.000	-.000	.000	-246.10	-167.0
2031	1.0060+07	-.000	-.000	.000	-250.78	-178.9
2032	1.0070+07	-.000	-.000	.000	-252.83	-160.2
2033	1.0080+07	-.000	-.000	.000	-255.64	-162.9
2034	1.0090+07	-.000	.000	.000	-257.49	179.8
2035	1.0100+07	-.000	-.000	.000	-259.81	-170.7

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

```
TPLOTF
 TPLOTF
 ENTER LOW, HIGH FREQUENCIES
9.9E6,10.1E6

     NSIZE =       21

  FREQUENCY (MHZ)            DECIBELS

        -170      -160      -150      -140      -130      -120
           I----+----I----+----I----+----I----+----I----+----I
   9.900 I
   9.910 I
   9.920 I
   9.930 I
   9.940 I
   9.950 I
   9.960 I
   9.970 I
   9.980 I
   9.990 I----------------------------
  10.000 I-------------------------------------------
  10.010 I----------------------------
  10.020 I
  10.030 I
  10.040 I
  10.050 I
  10.060 I
  10.070 I
  10.080 I
  10.090 I
  10.100 I
     FREQ
```

Figure 16. Spectrum at the Output of the First Bandpass Filter.

```
INPUT FORMAT,IDLMUL
 INPUT FORMAT,IDLMUL

 IDLMUL, ALO, FLO
 ALO = PEAK AMPLITUDE OF LO SIGNAL, VOLTS
 FLO = FREQ OF LO, HZ

IDLMUL,1.,9.E6
 IDLMUL,1.,9.E6
BLOCL←K,3
 BLOCK,3
 PROCESSING COMPLETE THRU BLOCK   3
TPLOTF
 TPLOTF
 ENTER LOW, HIGH FREQUENCIES
.9E6,1.1E6

      NSIZE =     21

  FREQUENCY (MHZ)          DECIBELS

       -180      -170      -160      -150      -140      -130
        I----+----I----+----I----+----I----+----I----+----I
   .900 I
   .910 I
   .920 I
   .930 I
   .940 I
   .950 I
   .960 I
   .970 I
   .980 I
   .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 17. Spectrum at the Output of the Ideal Multiplier.

```
PWBNDP, 1.E6, 20.E3, 4
 BWBNDP, 1.E6, 20.E3, 4
BLOCK, 4
 BLOCK, 4
 PROCESSING COMPLETE THRU BLOCK   4
TPLOTF
 TPLOTF
 ENTER LOW, HIGH FREQUENCIES
.9E6, 1.1E6

      NSIZE =      21

  FREQUENCY (MHZ)          DECIBELS

       -180      -170      -160      -150      -140      -130
         I----+----I----+----I----+----I----+----I----+----I
   .900  I
   .910  I
   .920  I
   .930  I
   .940  I
   .950  I
   .960  I
   .970  I
   .980  I
   .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 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

     NSIZE =      21

  FREQUENCY (MHZ)          DECIBELS

         -60       -50       -40       -30       -20       -10
          I----+----I----+----I----+----I----+----I----+----I
     .900 I
     .910 I
     .920 I
     .930 I
     .940 I
     .950 I
     .960 I
     .970 I
     .980 I
     .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.

```
AMDEMO
 AMDEMO
BLOCK,6
 BLOCK,6
 PROCESSING COMPLETE THRU BLOCK   6
TPLOTF
 TPLOTF
 ENTER LOW, HIGH FREQUENCIES
0.,200.E3

     NSIZE =      21

  FREQUENCY (KHZ)          DECIBELS

          -60       -50       -40       -30       -20       -10
           I----+----I----+----I----+----I----+----I----+----I
      .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
     FREQ
```

Figure 20. Spectrum at the Output of the AM Demodulator.

```
TPLOTT
 TPLOTT
 ENTER NSTART, NSTOP, NJUMP
1,2048,64
      +
              AMPLITUDE: MIN   .1875+00, MAX   .3125+00 VOLTS
      +
          0    .1   .2   .3   .4   .5   .6   .7   .8   .9  1.0
          I----I----I----I----I----I----I----I----I----I----I
       1  I      *
      65  I   *
     129  I *
     193  *
     257  *
     321  *
     385  I *
     449  I   *
     513  I      *
     577  I          *
     641  I              *
     705  I                   *
     769  I                        *
     833  I                             *
     897  I                                  *
     961  I                                      *
    1025  I                                          *
    1089  I                                             *
    1153  I                                               *
    1217  I                                                 *
    1281  I                                                 *
    1345  I                                                 *
    1409  I                                               *
    1473  I                                             *
    1537  I                                          *
    1601  I                                      *
    1665  I                                  *
    1729  I                             *
    1793  I                        *
    1857  I                   *
    1921  I              *
    1985  I          *
       N
```

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.

```
PRINTT
 PRINTT
 ENTER LOW, HIGH INDICES
1, 30

  LINE        REAL         IMAG

     1   2.058-01     0.000
     2   2.057-01     3.341-09
     3   2.055-01     3.147-09
     4   2.054-01     2.488-09
     5   2.053-01     1.673-09
     6   2.051-01     2.161-09
     7   2.050-01     2.205-09
     8   2.049-01     2.001-09
     9   2.047-01     1.950-09
    10   2.046-01     2.896-09
    11   2.045-01     2.416-09
    12   2.043-01     1.193-09
    13   2.042-01     1.673-09
    14   2.041-01     2.139-09
    15   2.039-01     1.892-09
    16   2.038-01     2.765-09
    17   2.037-01     1.455-09
    18   2.036-01     2.474-09
    19   2.034-01     2.387-09
    20   2.033-01     2.241-09
    21   2.032-01     1.426-09
    22   2.031-01     2.212-09
    23   2.029-01     1.892-09
    24   2.028-01     1.688-09
    25   2.027-01     9.022-10
    26   2.025-01     1.979-09
    27   2.024-01     1.310-09
    28   2.023-01     1.630-09
    29   2.022-01     1.513-09
    30   2.021-01     2.561-09
```

Figure 22. Listing of Time Function at the Output of the AM Demodulator.

```
TPLOTF
 TPLOTF
 ENTER LOW, HIGH FREQUENCIES
-1000-.E3,100.E3

     NSIZE =      21

  FREQUENCY (KHZ)           DECIBELS

          -60          -50          -40          -30          -20          -10
          I----+----I----+----I----+----I----+----I----+----I
 ******* I
 -90.000 I
 -80.000 I
 -70.000 I
 -60.000 I
 -50.000 I
 -40.000 I
 -30.000 I
 -20.000 I
 -10.000 I----------------------------
    .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
     FREQ
```

Figure 23. Spectrum at the Output of the AM Demodulator.

```
CHLOWP, 10.E3, 5, 1.

BLOCK, 7
 BLOCK, 7
 PROCESSING COMPLETE THRU BLOCK   7
TPLOTF
 TPLOTF
 ENTER LOW, HIGH FREQUENCIES
0., 200.E3

      NSIZE =      21

  FREQUENCY (KHZ)           DECIBELS

          -60         -50         -40         -30         -20        -10
           I----+----I----+----I----+----I----+----I----+----I
     .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
       FREQ




END OF JOB
 END OF JOB
END      16622 MLSEC
```

Figure 24. Spectrum at the Output of the Low Pass Filter.

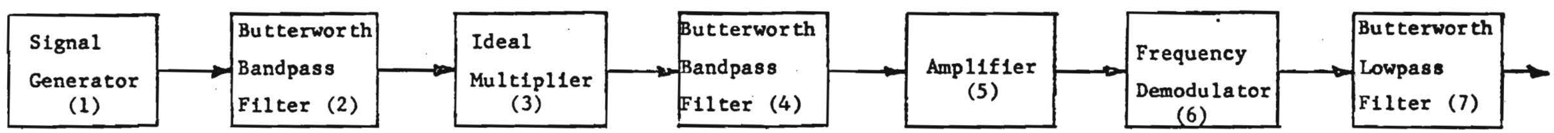


Figure 25. Block Diagram for FATCAT Analysis of an FM Receiver.

(4) A six-section intermediate frequency Butterworth bandpass filter with a center frequency of 1 MHz and a bandwidth of 80 kHz.

(5) An amplifier with a voltage gain of 140 dB.

(6) A frequency demodulator with a center frequency of 1 MHz.

(7) 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,

```
@XQT R.U
 START
SIGGEN, 1.E7, 1.E4, 0., 0., 30.E3, 1.E-6
 SIGGEN, 1.E7, 1.E4, 0., 0., 30.E3, 1.E-6
BLOCK, 1
 BLOCK, 1

 PERIOD =     1.000-04 SECONDS, DELTA-F =     1.000+04
 N =     2048, IGAM = 11 , DELTA-T =     4.883-08
 IS THIS SATISFACTORY
YES
 YES
 PROCESSING COMPLETE THRU BLOCK   1
TPLOTF
 TPLOTF
 ENTER LOW, HIGH FREQUENCIES
9.9E6, 10.1E6

     NSIZE =      21

  FREQUENCY (MHZ)           DECIBELS

      -180      -170      -160      -150      -140      -130
         I----+----I----+----I----+----I----+----I----+----I
  9.900  I
  9.910  I
  9.920  I
  9.930  I--
  9.940  I---------------
  9.950  I--------------------------
  9.960  I------------------------------------
  9.970  I-------------------------------------------
  9.980  I-----------------------------------------------
  9.990  I--------------------------------------------
 10.000  I------------------------------------------
 10.010  I--------------------------------------------
 10.020  I-----------------------------------------------
 10.030  I-------------------------------------------
 10.040  I------------------------------------
 10.050  I--------------------------
 10.060  I---------------
 10.070  I--
 10.080  I
 10.090  I
 10.100  I
    FREQ
```

Figure 26. Start of FATCAT Run for FM Receiver, Showing Spectrum at the Output of the Signal Generator.

```
BWBNDP, 1.E7,8Ø.E3, 1
 BWBNDP, 1.E7,8Ø.E3, 1
BLOCK, 2
 BLOCK, 2
 PROCESSING COMPLETE THRU BLOCK  2
TPLOTF,9.9E6, 1Ø.1E6
 TPLOTF,9.9E6, 1Ø.1E6

     NSIZE =      21

  FREQUENCY (MHZ)           DECIBELS

       -18Ø        -17Ø        -16Ø        -15Ø        -14Ø        -13Ø
          I----+----I----+----I----+----I----+----I----+----I
  9.9ØØ I
  9.91Ø I
  9.92Ø I
  9.93Ø I
  9.94Ø I---------
  9.95Ø I----------------------
  9.96Ø I---------------------------------
  9.97Ø I-----------------------------------------
  9.98Ø I----------------------------------------------
  9.99Ø I--------------------------------------------
 1Ø.ØØØ I------------------------------------------
 1Ø.Ø1Ø I--------------------------------------------
 1Ø.Ø2Ø I----------------------------------------------
 1Ø.Ø3Ø I-----------------------------------------
 1Ø.Ø4Ø I---------------------------------
 1Ø.Ø5Ø I----------------------
 1Ø.Ø6Ø I---------
 1Ø.Ø7Ø I
 1Ø.Ø8Ø I
 1Ø.Ø9Ø I
 1Ø.1ØØ I
     FREQ
```

Figure 27. Spectrum at the Output of the First Bandpass Filter.

```
IDLMUL, 1.,9.E6
 IDLMUL, 1.,9.E6
BLOCK, 3
 BLOCK, 3
 PROCESSING COMPLETE THRU BLOCK   3
TPLOTF,.9E6,1.1E6
 TPLOTF,.9E6,1.1E6

     NSIZE =     21

  FREQUENCY (MHZ)          DECIBELS

       -180      -170      -160      -150      -140      -130
          I----+----I----+----I----+----I----+----I----+----I
    .900  I
    .910  I
    .920  I
    .930  I
    .940  I---
    .950  I----------------
    .960  I---------------------------
    .970  I-----------------------------------
    .980  I----------------------------------------
    .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 28. Spectrum at the Output of the Ideal Multiplier.

```
BWBNDP, 1.E6,8.E4,6
 BWBNDP, 1.E6,8.E4,6
BLOV←CK,4
 BLOCK,4
 PROCESSING COMPLETE THRU BLOCK   4
TPLOTF,.9E6,1.1E6
 TPLOTF,.9E6,1.1E6

     NSIZE =      21

  FREQUENCY (MHZ)          DECIBELS

        -180        -170        -160        -150        -140        -130
           I----+----I----+----I----+----I----+----I----+----I
     .900  I
     .910  I
     .920  I
     .930  I
     .940  I
     .950  I----
     .960  I------------------------
     .970  I-----------------------------------
     .980  I----------------------------------------
     .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 29. Spectrum at the Output of the Second Bandpass Filter.

```
AMP, 140.
 AMP, 140.
BLOCK, 5
 BLOCK, 5
 PROCESSING  COMPLETE  THRU  BLOCK    5
TPLOTF, .9E6, 1.1E6
 TPLOTF, .9E6, 1.1E6

     NSIZE =      21

  FREQUENCY  (MHZ)              DECIBELS

          -40        -30        -20        -10          0         10
           I----+----I----+----I----+----I----+----I----+----I
     .900  I
     .910  I
     .920  I
     .930  I
     .940  I
     .950  I----
     .960  I------------------------
     .970  I-----------------------------------
     .980  I----------------------------------------
     .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 30. Spectrum at the Output of the Amplifier.

```
FMDEMO,1.E6
 FMDEMO,1.E6
BLOCK,6
 BLOCK,6
 PROCESSING COMPLETE THRU BLOCK  6
TPLOTF,0.,200.E3
 TPLOTF,0.,200.E3

     NSIZE =     21

  FREQUENCY (KHZ)          DECIBELS

         -40       -30       -20       -10        0        10
          I----+----I----+----I----+----I----+----I----+----I
     .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
     FREQ
```

Figure 31. Spectrum at the Output of the FM Demodulator.

```
TPLOTT
 TPLOTT
 ENTER NSTART, NSTOP, NJUMP
1,2048,64
      +
            AMPLITUDE: MIN -.2951+01, MAX  .2951+01 VOLTS
      +
         0   .1   .2   .3   .4   .5   .6   .7   .8   .9  1.0
         I----I----I----I----I----I----I----I----I----I----I
      1  I*
     65  I   *
    129  I      *
    193  I         *
    257  I             *
    321  I                 *
    385  I                      *
    449  I                           *
    513  I                               *
    577  I                                    *
    641  I                                        *
    705  I                                           *
    769  I                                              *
    833  I                                                *
    897  I                                                 *
    961  I                                                 *
   1025  I                                                *
   1089  I                                             *
   1153  I                                          *
   1217  I                                       *
   1281  I                                   *
   1345  I                               *
   1409  I                          *
   1473  I                     *
   1537  I                 *
   1601  I            *
   1665  I        *
   1729  I     *
   1793  I  *
   1857  I*
   1921  *
   1985  *
      N
```

Figure 32. Time Waveform at the Output of the FM Demodulator.

```
PRINTF
 PRINTF
 ENTER LOW, HIGH FREQUENCIES
0.,200.E3
```

LINE	FREQ	REAL	IMAG	MAG	DB	PHASE
1025	0.0000	-.000	.000	.000	-164.10	180.0
1026	1.0000+04	-1.384	-.448	1.454	3.25	-162.1
1027	2.0000+04	.000	.000	.000	-108.66	28.8
1028	3.0000+04	-.000	-.020	.020	-33.92	-90.0
1029	4.0000+04	.000	-.000	.000	-110.85	-44.1
1030	5.0000+04	-.001	-.014	.014	-37.08	-92.3
1031	6.0000+04	-.000	.000	.000	-117.33	130.8
1032	7.0000+04	-.007	-.001	.007	-43.44	-168.6
1033	8.0000+04	.000	-.000	.000	-124.88	-17.6
1034	9.0000+04	.000	.001	.001	-62.81	82.4
1035	1.0000+05	-.000	-.000	.000	-138.68	-122.5
1036	1.1000+05	-.000	-.000	.000	-72.29	-132.0
1037	1.2000+05	-.000	.000	.000	-150.36	107.8
1038	1.3000+05	-.000	.000	.000	-78.32	145.4
1039	1.4000+05	.000	-.000	.000	-154.60	-40.4
1040	1.5000+05	.000	.000	.000	-91.80	53.2
1041	1.6000+05	-.000	-.000	.000	-164.34	-160.7
1042	1.7000+05	-.000	-.000	.000	-107.43	-140.9
1043	1.8000+05	.000	.000	.000	-174.48	47.2
1044	1.9000+05	-.000	.000	.000	-110.12	104.0
1045	2.0000+05	-.000	.000	.000	-192.12	100.0

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

  FREQUENCY (KHZ)          DECIBELS

          -40        -30        -20        -10          0         10
           I----+----I----+----I----+----I----+----I----+----I
      .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
        FREQ
```

Figure 34. Spectrum at the Output of the Low Pass Filter.

```
TPLOTT
 TPLOTT
 ENTER NSTART, NSTOP, NJUMP
1,2048,64
      +
          AMPLITUDE: MIN -.2895+01, MAX  .2895+01 VOLTS
      +
        0   .1   .2   .3   .4   .5   .6   .7   .8   .9  1.0
        I----I----I----I----I----I----I----I----I----I----I
     1  I                                    *
    65  I                                        *
   129  I                                           *
   193  I                                              *
   257  I                                                *
   321  I                                                 *
   385  I                                                 *
   449  I                                                *
   513  I                                              *
   577  I                                           *
   641  I                                        *
   705  I                                    *
   769  I                               *
   833  I                          *
   897  I                     *
   961  I                 *
  1025  I            *
  1089  I        *
  1153  I     *
  1217  I  *
  1281  I*
  1345  *
  1409  *
  1473  I*
  1537  I  *
  1601  I     *
  1665  I        *
  1729  I            *
  1793  I                 *
  1857  I                      *
  1921  I                           *
  1985  I                               *
     N
```

Figure 35. Time Waveform at the Output of the Low Pass Filter.

```
PRINTF
 PRINTF
 ENTER LOW, HIGH FREQUENCIES
0.,200.E3
```

LINE	FREQ	REAL	IMAG	MAG	DB	PHASE
1025	0.0000	-.000	-.000	.000	-163.69	-180.0
1026	1.0000+04	.691	-1.280	1.454	3.25	-61.6
1027	2.0000+04	-.000	.000	.000	-133.95	147.3
1028	3.0000+04	.000	-.000	.000	-94.12	-81.1
1029	4.0000+04	-.000	-.000	.000	-170.51	-160.3
1030	5.0000+04	-.000	-.000	.000	-140.61	-160.1
1031	6.0000+04	.000	.000	.000	-175.73	79.9
1032	7.0000+04	.000	-.000	.000	-161.78	-34.8
1033	8.0000+04	.000	-.000	.000	-182.63	-14.2
1034	9.0000+04	-.000	-.000	.000	-169.67	-153.3
1035	1.0000+05	-.000	-.000	.000	-181.84	-170.3
1036	1.1000+05	.000	.000	.000	-168.75	51.7
1037	1.2000+05	.000	.000	.000	-176.93	54.6
1038	1.3000+05	-.000	.000	.000	-167.36	159.3
1039	1.4000+05	.000	-.000	.000	-182.79	-46.1
1040	1.5000+05	.000	-.000	.000	-168.67	-27.5
1041	1.6000+05	-.000	-.000	.000	-187.43	-116.5
1042	1.7000+05	-.000	-.000	.000	-160.39	-150.5
1043	1.8000+05	.000	.000	.000	-183.37	49.8
1044	1.9000+05	.000	.000	.000	-182.41	72.0
1045	2.0000+05	-.000	-.000	.000	-180.54	-98.5

```
END OF JOB
 END OF JOB
END     19203 MLSEC
```

Figure 36. 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 <u>subprogram</u> 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 $\ddot{x} + \dot{x} + x = 0$ 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 $-\ddot{x} = x + \dot{x}$, 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 $\ddot{x}$ to give $\dot{x}$ 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

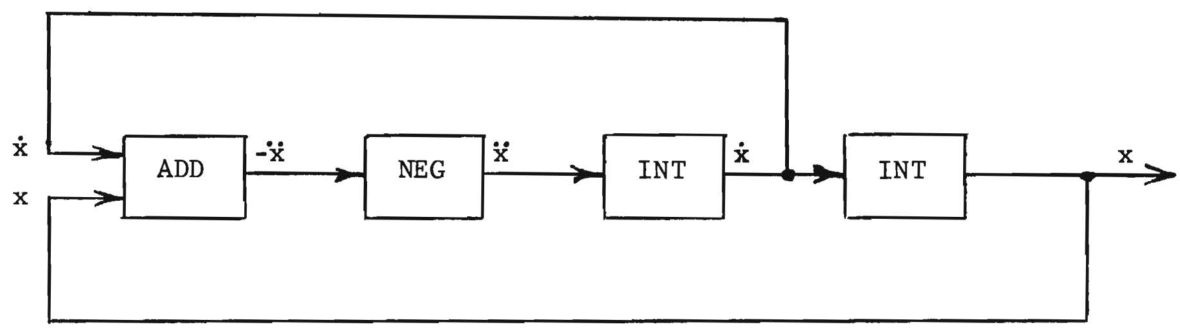


Figure 37. Detailed MIMIC Block Diagram for a System Described by $\ddot{x} + \dot{x} + x = 0$.

10	19
NEG2DX	ADD(X,1DX)
2DX	NEG(NEG2DX)
1DX	INT(2DX,0.)
X	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, K2, in the differential equation, the instruction "CON(K2)", 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 K2 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

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

and the associated differential equation is

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

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

```
X1        INT(K * U1 - A * X1, 0.) .                    (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:

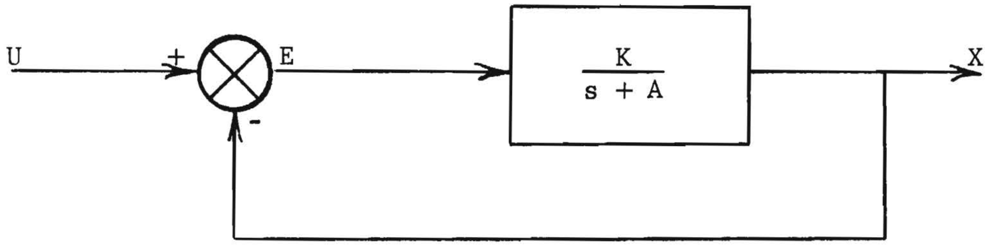


Figure 39. Block Diagram of Feedback Example.

```
@GT*LIB.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***
  TIME          X
 0.00000       0.00000                2.6000          .85703
 1.00000-01    9.46962-02             2.7000          .86245
  .20000        .17953                2.8000          .86731
  .30000        .25552                2.9000          .87166
  .40000        .32360                3.0000          .87556
  .50000        .38459                3.1000          .87905
  .60000        .43923                3.2000          .88218
  .70000        .48817                3.3000          .88499
  .80000        .53202                3.4000          .88750
  .90000        .57129                3.5000          .88975
 1.00000        .60648                3.6000          .89176
 1.1000         .63800                3.7000          .89357
 1.2000         .66624                3.8000          .89518
 1.3000         .69154                3.9000          .89663
 1.4000         .71420                4.0000          .89793
 1.5000         .73450                4.1000          .89909
 1.6000         .75269                4.2000          .90013
 1.7000         .76898                4.3000          .90107
 1.8000         .78357                4.4000          .90190
 1.9000         .79665                4.5000          .90265
 2.0000         .80836                4.6000          .90332
 2.1000         .81885                4.7000          .90392
 2.2000         .82825                4.8000          .90446
 2.3000         .83667                4.9000          .90494
 2.4000         .84422                5.0000          .90538
 2.5000         .85097                5.1000          .90576
                                    END       568 MLSEC
```

Figure 40. MIMIC Simulation of System of Figure 39.

```
10          19
FIRST       BSP(U1, K, A)
X1          INT(K*U1 - A*X1, 0.)
FIRST       ESP(X1)                      (4)
```

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.

```
@GT*LIB.MIMIC,IS TEST
MIMIC-03.2-06/13/72-08:33:42

@ADD LARRY.4
     1*                                PAR(X)
     2*               FIRST            BSP(U1,K,A)
     3*               X1               INT(K*U1-A*X1,0.)
     4*               FIRST            ESP(X1)
     5*               E                1.-X
     6*               FIRST            CSP(E,1.,.1)
     7*                                RSP(X)
     8*                                FIN(T,5.)
     9*                                HDR(TIME,E,X)
    10*                                OUT(T,E,X)
    11*                                END

**FURTHER DIAGNOSTICS AND EXECUTION FOLLOW***

ENTER DATA NOW
     X
   0.00000
```

Figure 41. Subprogram Example.

C. TIMSIM: A User-Oriented Communications System Block Diagram Simulation Program

TIMSIM is not a new digital simulation language to compete with MIMIC, nor is it really a computer program. Rather, it is an operating philosphy 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

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

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

$$\frac{X}{U} = \frac{sK}{s^2 \, (f_o/\omega_o B) + s + (\omega_o f_o/B)} \qquad (6)$$

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

$$X = \frac{1}{s}\left[2\pi B(KU - X) - \frac{1}{s}(\omega_o^2 X)\right] \tag{7}$$

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

$$\begin{aligned} \dot{x} &= 2\pi B(Ku - x) + y \\ \dot{y} &= -\omega_o^2\, x. \end{aligned} \tag{8}$$

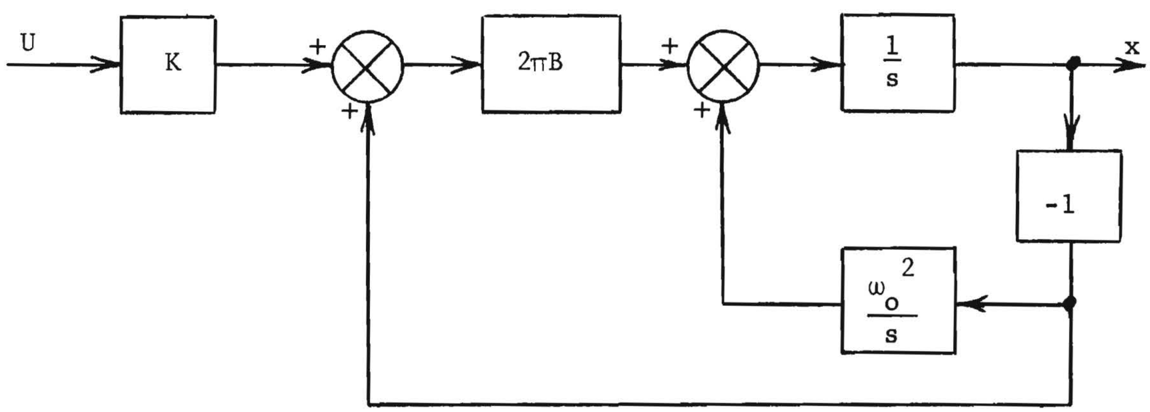


Figure 42. Single-Tuned Filter Flow Diagram.

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: $x \rightarrow$ OUT110; $y \rightarrow$ Y110; $u \rightarrow$ XIN110; $K \rightarrow$ K110; $B \rightarrow$ B110; $f_o \rightarrow$ FRQ110.

2. TIMSIM Subprogram Library.

Table I contains the subprograms used to model AM signal generators (AMSIG, AMSIG5), 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

TIMSIM SUBPROGRAM LIBRARY

```
****BEGIN SUBPROGRAM FOR AMPLITUDE MODULATED SIGNAL GENERATOR.***
          AMSIG     BSP(XIN100,FRQ100,M100,K100)
          THT100    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    BSP(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     BSP(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)

```
****BEGIN SUBPROGRAM FOR MIXER****
          MIXER     BSP(XIN120,FRQ120,K120)
          OUT120    XIN120*K120*SIN(6.2831*FRQ120*T)
          MIXER     ESP(OUT120)
****END OF SUBPROGRAM****
```

Definitions

```
          XIN120 = Input Voltage
          FRQ120 = Mixer Product Frequency
          K120   = Gain
          OUT120 = Output Voltage
```

```
****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****
```

Definitions

```
          XIN130 = Input Voltage
          FCN130 = Corner Frequency, Hz
          K130   = DC Gain
          OUT130 = Output Voltage
```

```
****BEGIN SUBPROGRAM FOR RC BANDPASS FILTER****
          BPFIL     BSP(XIN140,FL140,FH140,K140)
          WL140     6.2831*FL140
          WH140     6.2831*FH140
          Y140      INT(WL140*WH140*OUT140,0.)
          OUT140    INT(K140*WH140*XIN140-(WL140+WH140)*OUT140-Y140,0.)
          BPFIL     ESP(OUT140)
****END OF SUBPROGRAM****
```

Definitions

```
          XIN140 = Input Voltage
          FL140  = Lower Corner Frequency, Hz
          FH140  = Higher Corner Frequency, Hz
          K140   = Center Frequency Gain
          OUT140 = Output Voltage
```

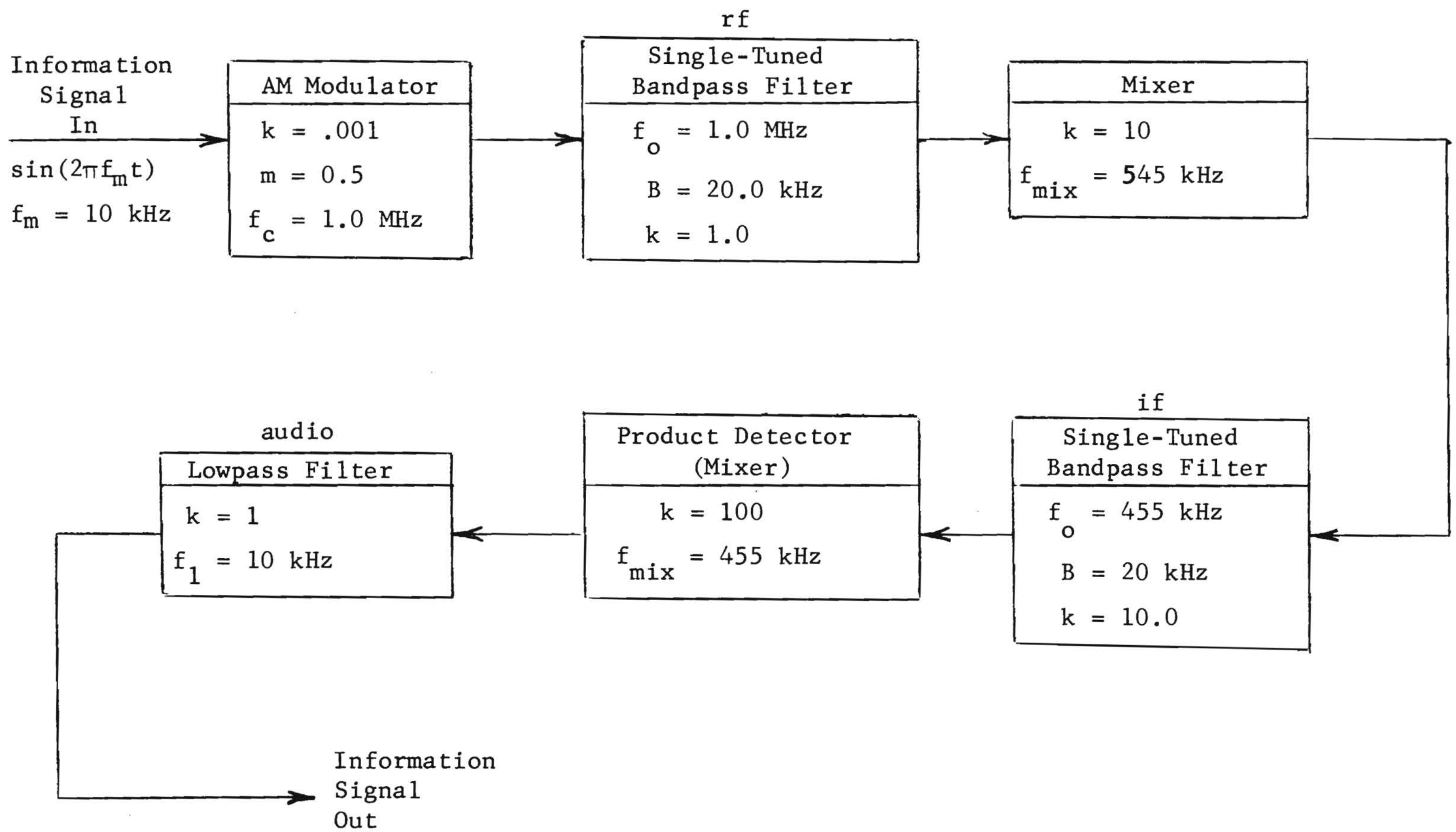


Figure 43. Hypothetical AM Communication System.

```
*                         CON(FM,FC,M,K,FO,B)
*                         CON(FMIX,KMIX)
*                         CON(F01,B01,K01)
*                         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.+M100*XIN100)*COS(THT100)
*              AMSIG      ESP(OUT100)
*****END OF SUBPROGRAM AMSIG*********************
*****BEGIN SUBPROGRAM FOR SINGLE-TUNED FILTER********
*              STFIL      BSP(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 STFIL*********
*****BEGIN SUBPROGRAM FOR MIXER******************
*              MIXER      BSP(XIN120,FRQ120,K120)
*              OUT120     XIN120*K120*SIN(6.2831*FRQ120*T)
*              MIXER      ESP(OUT120)
*****END OF SUBPROGRAM MIXER*********************
*****BEGIN SUBPROGRAM FOR SINGLE-TUNED FILTER*********
*              STFIL1     BSP(XIN111,B111,FRQ111,K111)
*              OUT111     INT(Y111+(6.2831*B111)*(K111*XIN111-OUT111),0.)
*              Y111       INT(-(6.2831*FRQ111)*(6.2831*FRQ111)*OUT111,0.)
*              STFIL1     ESP(OUT111)
*****END OF SUBPROGRAM STFIL1*********
*****BEGIN SUBPROGRAM FOR MIXER**********************
*              MIXER1     BSP(XIN121,FRQ121,K121)
*              OUT121     XIN121*K121*SIN(6.2831*FRQ121*T)
*              MIXER1     ESP(OUT121)
*****END OF SUBPROGRAM MIXER1*************************
*****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 LPFIL********************
*              X          SIN(6.2831*FM*T)
*              AMSIG      CSP(X,FC,M,K)
*                         RSP(Y)
*              STFIL      CSP(Y,B,FO,1.)
*                         RSP(Z)
*              MIXER      CSP(Z,FMIX,KMIX)
*                         RSP(ZZ)
*              STFIL1     CSP(ZZ,B01,F01,K01)
*                         RSP(ZZZ)
*              MIXER1     CSP(ZZZ,FMIX1,KMIX1)
*                         RSP(OUTPUT)
*              LPFIL      CSP(OUTPUT,FLP,KLP)
*                         RSP(V)
```

Figure 44. TIMSIM Listing, Hypothetical Communication System. (Continued)

```
                DT1       1./(50.*FM)
                DT2       1./(20.*FC)
                T1        (20./(4.*FM))-DT2
                TF        2./FM
                TTEST     T-T1
                TLOGIC    FSW(TTEST,TRUE,FALSE,FALSE)
                NTLOG     COM(TLOGIC)
    TLOGIC      DT        EQL(DT1)
    NTLOG       DT        EQL(DT2)
                          FIN(T,TF)
                          HDR(TIME,X,V)
                          OUT(T,X,V)
                          END

 **FURTHER DIAGNOSTICS AND EXECUTION FOLLOW***


 ENTER DATA NOW

    FM          FC             M          K             FO            B
    10000.      1.00000+06     .50000     1.00000-03    1.00000+06    20000.


 ENTER DATA NOW

    FMIX           KMIX
    5.45000+05     10.000


 ENTER DATA NOW

    FO1            BO1        KO1
    4.55000+05     20000.     10.000


 ENTER DATA NOW

    FMIX1          KMIX1
    4.55000+05     100.00


 ENTER DATA NOW

    FLP       KLP
    10000.    1.0000
```

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 STFIL1 and the internal variable names terminate in 111 rather than in 110. 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 FO, 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 DT1 is used for T < T1 while DT2 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.

TIME	X	V
0.00000	0.00000	0.00000
2.00000-06	.12533	-9.69228-04
4.00000-06	.24869	-6.16795-03
6.00000-06	.36812	-1.71452-02
8.00000-06	.48175	-3.61954-02
1.00000-05	.58778	-6.83026-02
1.20000-05	.68454	-.11594
1.40000-05	.77051	-.17394
1.60000-05	.84432	-.23469
1.80000-05	.90482	-.29958
2.00000-05	.95105	-.38015
2.20000-05	.98228	-.48357
2.40000-05	.99803	-.59961
2.60000-05	.99803	-.70867
2.80000-05	.98229	-.80481
3.00000-05	.95106	-.90508
3.20000-05	.90484	-1.0299
3.40000-05	.84434	-1.1756
3.60000-05	.77053	-1.3140
3.80000-05	.68457	-1.4232
4.00000-05	.58781	-1.5143
4.20000-05	.48179	-1.6194
4.40000-05	.36816	-1.7525
4.60000-05	.24873	-1.8900
4.80000-05	.12537	-1.9960
5.00000-05	4.29471-05	-2.0634
5.20000-05	-.12529	-2.1229
5.40000-05	-.24864	-2.2064
5.60000-05	-.36808	-2.3090
5.80000-05	-.48171	-2.3936
6.00000-05	-.58774	-2.4340
6.20000-05	-.68451	-2.4449
6.40000-05	-.77048	-2.4639
6.60000-05	-.84430	-2.5082
6.80000-05	-.90480	-2.5549
7.00000-05	-.95104	-2.5694
7.20000-05	-.98228	-2.5453
7.40000-05	-.99802	-2.5109
7.60000-05	-.99803	-2.4963
7.80000-05	-.98230	-2.4999
8.00000-05	-.95108	-2.4928
8.20000-05	-.90486	-2.4537
8.40000-05	-.84437	-2.3932
8.60000-05	-.77056	-2.3409
8.80000-05	-.68460	-2.3125
9.00000-05	-.58785	-2.2943
9.20000-05	-.48182	-2.2608
9.40000-05	-.36820	-2.2047
9.60000-05	-.24877	-2.1453
9.80000-05	-.12542	-2.1069
1.00000-04	-8.59539-05	-2.0929

Figure 45. TIMSIM Results, Hypothetical AM Communication System. (Continued)

1.02000-04	.12525	-2.0830
1.04000-04	.24860	-2.0566
1.06000-04	.36804	-2.0173
1.08000-04	.48167	-1.9894
1.10000-04	.58771	-1.9916
1.12000-04	.68448	-2.0153
1.14000-04	.77045	-2.0339
1.16000-04	.84427	-2.0326
1.18000-04	.90478	-2.0267
1.20000-04	.95102	-2.0456
1.22000-04	.98227	-2.0982
1.24000-04	.99802	-2.1612
1.26000-04	.99803	-2.2036
1.28000-04	.98231	-2.2228
1.30000-04	.95109	-2.2486
1.32000-04	.90488	-2.3094
1.34000-04	.84439	-2.3969
1.36000-04	.77059	-2.4735
1.38000-04	.68463	-2.5141
1.40000-04	.58788	-2.5358
1.42000-04	.48186	-2.5782
1.44000-04	.36824	-2.6564
1.46000-04	.24881	-2.7415
1.48000-04	.12546	-2.7927
1.50000-04	1.29795-04	-2.8040
1.52000-04	-.12520	-2.8107
1.54000-04	-.24856	-2.8478
1.56000-04	-.36800	-2.9085
1.58000-04	-.48163	-2.9521
1.60000-04	-.58767	-2.9509
1.62000-04	-.68444	-2.9216
1.64000-04	-.77042	-2.9045
1.66000-04	-.84425	-2.9169
1.68000-04	-.90476	-2.9340
1.70000-04	-.95101	-2.9193
1.72000-04	-.98226	-2.8665
1.74000-04	-.99802	-2.8058
1.76000-04	-.99804	-2.7679
1.78000-04	-.98232	-2.7506
1.80000-04	-.95110	-2.7236
1.82000-04	-.90489	-2.6652
1.84000-04	-.84441	-2.5864
1.86000-04	-.77062	-2.5178
1.88000-04	-.68467	-2.4750
1.90000-04	-.58792	-2.4436
1.92000-04	-.48190	-2.3973
1.94000-04	-.36828	-2.3291
1.96000-04	-.24886	-2.2585
1.98000-04	-.12550	-2.2105
2.00000-04	-1.74173-04	-2.1878
2.02000-04	.12516	-2.1696

END 39256 MLSEC

Figure 45. End.

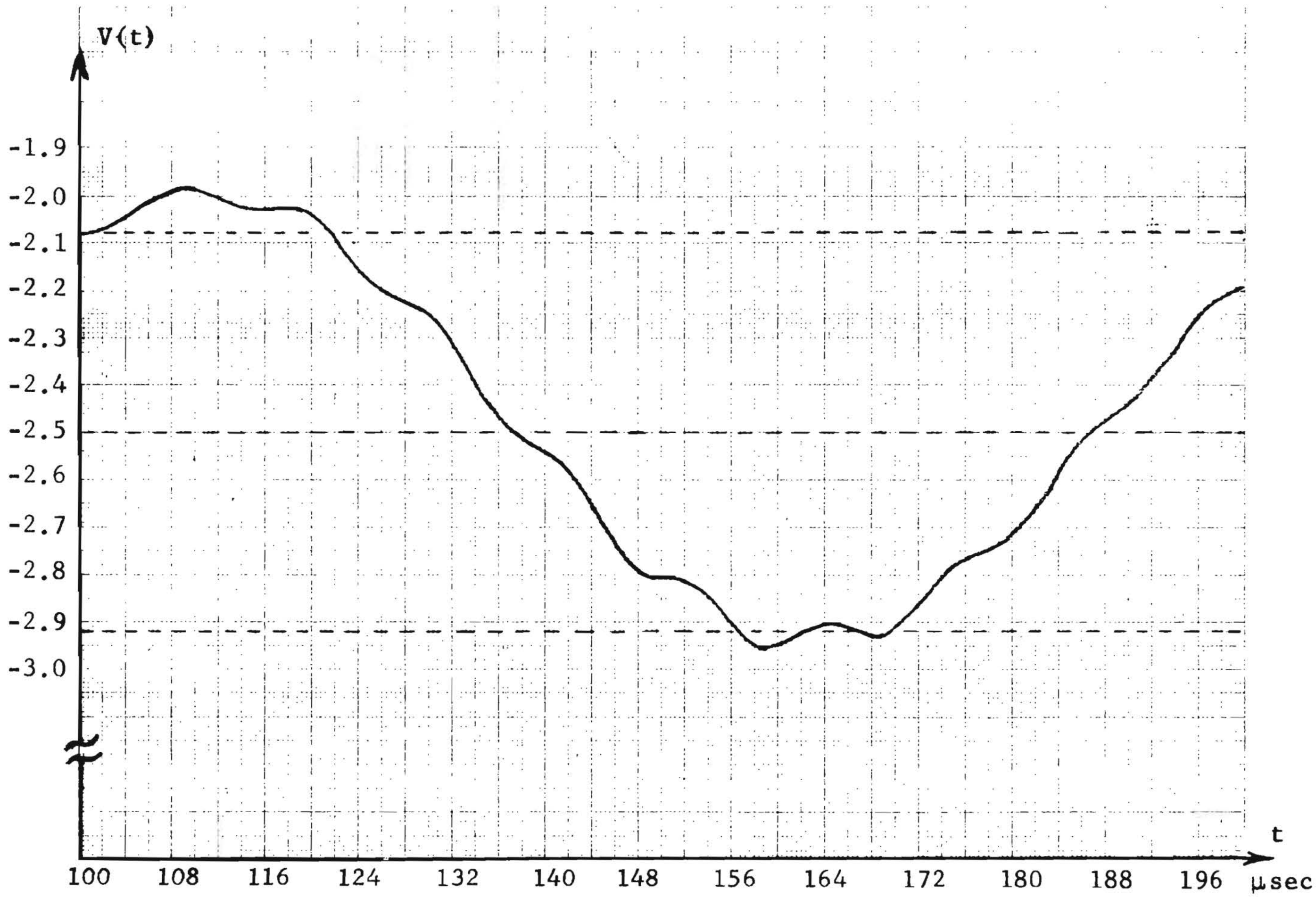


Figure 46. Plot of TIMSIM Results for AM System.

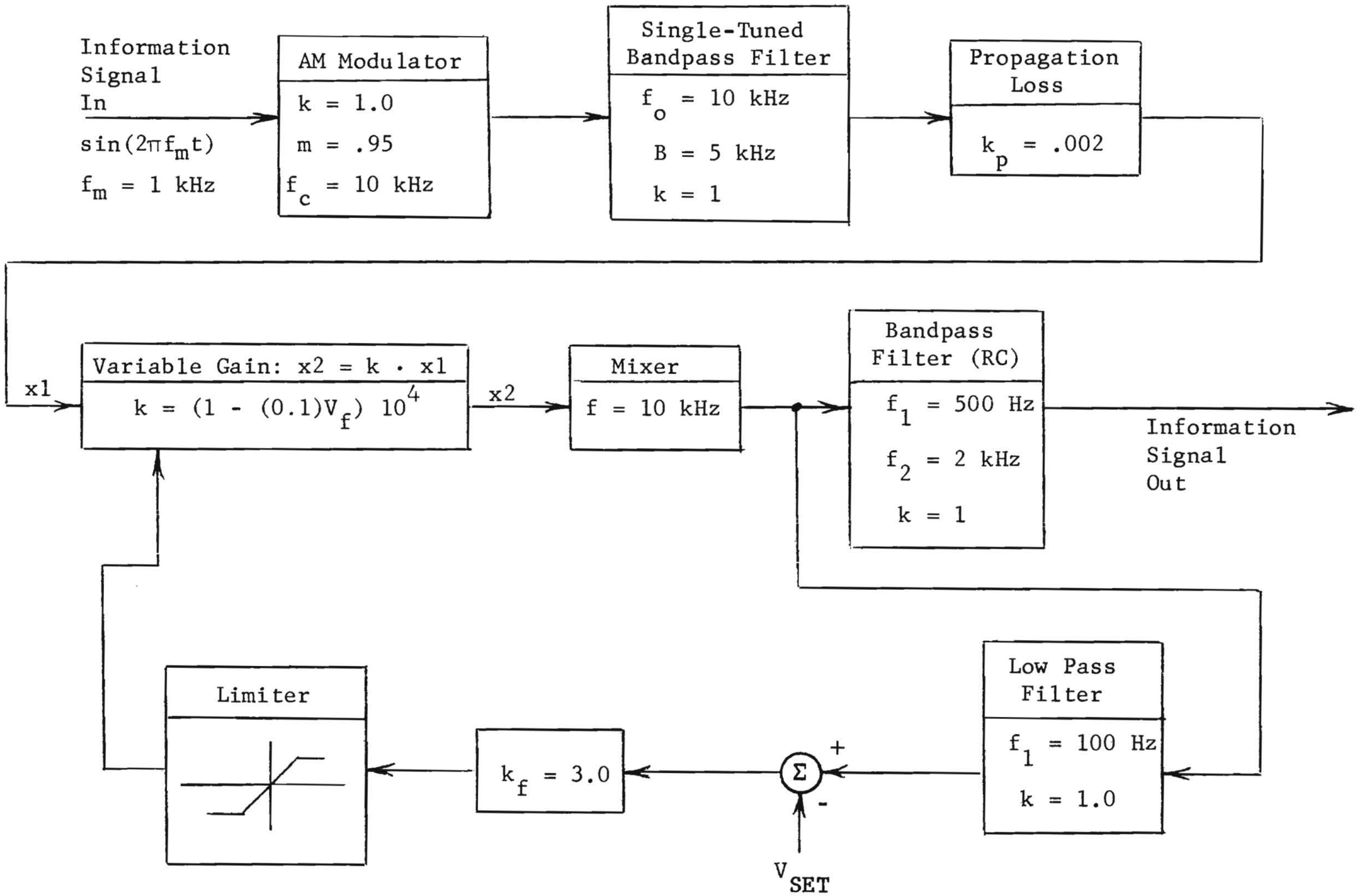


Figure 47. Hypothetical Automatic Gain Control System.

```
 1*                         CON(FC,KP,BW,K1,VSET)
 2*                         PAR(X4,FCR,KF,TPOINT,TF,KFORWD)
 3*                         PAR(FM)
 4*****BEGIN SUBPROGRAM FOR LOW-PASS FILTER****
 5*             LPFIL       BSP(XIN130,FCN130,K130)
 6*             OUT130      INT(6.2831*FCN130*(XIN130-OUT130),0.)
 7*             LPFIL       ESP(OUT130)
 8***
     **END OF SUBPROGRAM***
 9*****BEGIN SUBPROGRAM FOR SINGLE-TUNED FILTER****
10*             STFIL       BSP(XIN110,B110,FRQ110,K110)
11*             OUT110      INT(Y110+(6.2831*B110)*(K110*XIN110-OUT110),0.)
12*             Y110        INT(-(6.2831*FRQ110)*(6.2831*FRQ110)*OUT110,0 .)
13*             STFIL       ESP(OUT110)
14****
      *
      END OF SUBPROGRAM****
15*****BEGIN SUBPROGRAM FOR SIN AMPLITUDE MODULATED SIGNAL GENERATOR.
   ***
16*             AMSIG5      BSP(XIN105,FRQ105,M105,K105)
17*             THT105      6.2831*FRQ105*T
18*             OUT105      K105*(1.+M105*XIN105)*SIN(THT105)
19*             AMSIG5      ESP(OUT105)
20*****END OF SUBPROGRAM****
21*
   *
    ***BEGIN SUBPROGRAM FOR RC BANDPASS FILTER****
22*             BPFIL       BSP(XIN140,FL140,FH140,K140)
23*             WL140       6.2831*FL140
24*             WH140       6.2831*FH140
25*             Y140        INT(WL140*WH140*OUT140,0.)
26*             OUT140      INT(K140*WH140*XIN140-(WL140+WH140)*OUT140-Y140,0.)
27*             BPFIL       ESP(OUT140)
28*****END OF SUBPROGRAM****
29*             XIN         SIN(6.2831*FM*T)
30*             AMSIG5      CSP(XIN,FC,.95,1.)
31*                         RSP(XX)
32*             STFIL       CSP(XX,BW,FC,1.)
33*                         RSP(X)
34*             X1          KP*X
35*             KVAR        1. - K1*VF
36*             X2          KVAR*X1*KFORWD
37*             X3          X2*SIN(6.2831*FC*T)
38*             BPFIL       CSP(X3,.5*FM,.2*FC,1.)
39*                         RSP(X5)
40*             LPFIL       CSP(X3,FCR,1.)
41*                         RSP(X4)
42*             VF1         KF*(X4-VSET)
43*             VFLIM       .9/K1
44*             VF          LIM(VF1,-VFLIM,VFLIM)
```

Figure 48. TIMSIM Listing, Automatic Gain Control System. (Continued).

```
                DT1          0.05/FCR
                DT2          .50/FC
                T1           TPOINT - DT2
                TTEST        T-T1
                TLOGIC       FSW(TTEST,TRUE,FALSE,FALSE)
                NTLOG        COM(TLOGIC)
     TLOGIC     DT           EQL(DT1)
     NTLOG      DT           EQL(DT2)
                             FIN(T,TF)
                             HDR(TIME,X3,X4,VF,X5)
                             OUT(T,X3,X4,VF,X5)
                             END

 **FURTHER DIAGNOSTICS AND EXECUTION FOLLOW***


 ENTER DATA NOW
    FC          KP             BW           K1               VSET
    10000.      2.00000-03     5000.0       1.00000-01       1.0000


 ENTER DATA NOW
    0.        100.         3.            4.E-3          6.E-3         1.E4
   X4         FCR         KF          TPOINT           TF            KFORWD
  0.00000     100.00      3.0000     4.00000-03      6.00000-03     10000.


 ENTER DATA NOW
 1000.
    FM
    1000.0
```

Figure 48. End.

TIME	X3	X4	VF	X5
0.00000	0.00000	0.00000	-3.0000	0.00000
5.00000-04	-4.07690-05	2.8968	5.6903	-3.42534-02
1.00000-03	1.02872-04	2.6581	4.9742	-1.5398
1.50000-03	-6.78247-05	3.4256	7.2767	-.38317
2.00000-03	1.84120-04	2.9041	5.7124	-.63602
2.50000-03	-9.51716-05	3.4685	7.4054	-1.71602-02
3.00000-03	2.84755-04	2.9241	5.7723	-.47952
3.50000-03	-1.15230-04	3.4720	7.4159	3.04278-02
4.00000-03	3.95770-04	2.9258	5.7773	-.46301
4.05000-03	3.01825-04	2.9397	5.8191	.34545
4.10000-03	1.84063-04	2.9858	5.9573	1.1311
4.15000-03	6.21273-05	3.0570	6.1709	1.7666
4.20000-03	-4.57693-05	3.1427	6.4279	2.1556
4.25000-03	-1.27702-04	3.2317	6.6947	2.2567
4.30000-03	-1.79128-04	3.3140	6.9417	2.0872
4.35000-03	-2.00990-04	3.3825	7.1474	1.7065
4.40000-03	-1.97042-04	3.4329	7.2985	1.1917
4.45000-03	-1.71074-04	3.4630	7.3889	.61588
4.50000-03	-1.25771-04	3.4723	7.4168	3.57353-02
4.55000-03	-6.23912-05	3.4610	7.3829	-.51181
4.60000-03	1.81439-05	3.4298	7.2894	-1.0048
4.65000-03	1.14925-04	3.3801	7.1404	-1.4290
4.70000-03	2.24564-04	3.3142	6.9426	-1.7688
4.75000-03	3.39990-04	3.2358	6.7074	-2.0009
4.80000-03	4.49451-04	3.1508	6.4525	-2.0913
4.85000-03	5.37047-04	3.0675	6.2026	-2.0006
4.90000-03	5.85543-04	2.9958	5.9874	-1.6962
4.95000-03	5.81355-04	2.9459	5.8376	-1.1710
5.00000-03	5.20045-04	2.9259	5.7778	-.46135
5.05000-03	4 09642-04	2.9399	5.8195	.34709
5.10000-03	2.68857-04	2.9859	5.9577	1.1327
5.15000-03	1.21864-04	3.0571	6.1712	1.7682
5.20000-03	-9.49593-06	3.1428	6.4282	2.1572
5.25000-03	-1.10613-04	3.2317	6.6950	2.2584
5.30000-03	-1.75682-04	3.3140	6.9419	2.0888
5.35000-03	-2.05506-04	3.3826	7.1476	1.7080
5.40000-03	-2.04383-04	3.4329	7.2987	1.1931
5.45000-03	-1.76745-04	3.4630	7.3891	.61719
5.50000-03	-1.25781-04	3.4723	7.4170	3.69046-02
5.55000-03	-5.30914-05	3.4610	7.3830	-.51079
5.60000-03	4.04190-05	3.4299	7.2896	-1.0039
5.65000-03	1.53555-04	3.3802	7.1405	-1.4282
5.70000-03	2.82489-04	3.3142	6.9427	-1.7683
5.75000-03	4.19098-04	3.2358	6.7075	-2.0004
5.80000-03	5.49811-04	3.1509	6.4527	-2.0910
5.85000-03	6.56145-04	3.0676	6.2027	-2.0004
5.90000-03	7.17897-04	2.9958	5.9875	-1.6960
5.95000-03	7.18865-04	2.9459	5.8377	-1.1707
6.00000-03	6.53252-04	2.9260	5.7779	-.46095
6.05000-03	5.29630-04	2.9399	5.8196	.34764

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

1. Holland, L. D., J. R. Walsh and R. D. Wetherington, Communication System Modeling, Technical Report No. 9, Contract No. NAS8-20054, Georgia Institute of Technology, Engineering Experiment Station, 19 November 1971.

2. Walsh, J. R., R. D. Wetherington and L. D. Holland, Circuit Detail Modeling of the Airlock Module Transmitter, Technical Report No. 10, Contract No. NAS8-20054, Georgia Institute of Technology, Engineering Experiment Station, 19 November 1971.

3. Milliman, L. D., W. A. Massena and R. H. Dickhaut, A Digital Computer Program for Transient Analysis of Electronic Circuits Users Guide, Report 346-1, Contract No. DA-49-186-AMC-346(X), The Boeing Company, Seattle, Washington, January 1967.

4. Stephenson, R. E., Computer Simulation for Engineers, Harcourt, Brace, Jovanovich, Inc., New York, N. Y., 1971, pp 163-165.

5. Peterson, H. E., F. J. Sanson and L. M. Warshawsky, MIMIC - A Digital Simulation Program, SESCA Internal Memo 65-12, Directorate for Computation, Deputy for Studies and Analysis, Systems Engineering Group, Wright-Patterson AFB, Ohio, May 1965.

6. Pettit, J. M , and M. M. McWhorter, Electronic Amplifier Circuits, McGraw-Hill, New York, N. Y., 1961, p 168.

7. Ogata, K., State Space Analysis of Control Systems, Prentice-Hall, Inc., Englewood Cliffs, N. J., 1967, p 182.

APPENDIX A

Listing of Software Modification to CIRCUS to Write Output Data into External File.

```
      SUBROUTINE LINK6A(PLOT,NPNT,TIME,DUMY2,LFLAG,DUMY3,NWP,
     1                  DELT,NREAD )
      INTEGER HOUR(2), DAY(2)
      DIMENSION WORD(1), PLOT(NPNT,1), TIME(1)
      DIMENSION NWORD(1)
      COMMON WORD,N1,N5,N6,LPDS,LDS,JTIILE(192)
      COMMON INDPLT,IPLOTI,NPLOTS,KPRNT,DUMY1,DTCUR
      COMMON /SCRTCH/ TITLE(100),AMAX,AMIN,DAY,HOUR,IBAD,IMAX,
     1        IMIN,KK,LA,NGO,NN,J,NWIPE,PI1,I1,I2,I3,I4,
     2        XPLOTS, A
      EQUIVALENCE ( WORD(126), N7 ), ( WORD(38), N2 )
      EQUIVALENCE ( WORD(33),LINTVL )
      EQUIVALENCE ( WORD,NWORD )
      EQUIVALENCE ( WORD(109),LYMAX   )
      EQUIVALENCE ( WORD(110),LYMIN   )
      EQUIVALENCE ( WORD(111),NPCELL )
      EQUIVALENCE ( WORD(112),LPLT    )
      NR = 51
      NC = 21
      IF ( LFLAG-1 ) 390,385,390
  385 CONTINUE
      LFLAG = 2
      WRITE (N7) ( PLOT(I,1), I=1,NWIPE )
      NREAD = NWIPE
      REWIND N7
      READ (N2) XPLOTS, (TITLE(I),I=1,NPLOTS)
  390 CONTINUE
      WRITE (N6,30)
C**** READ TIME AND PLOT VARIABLES FOR NPNI VALUES OF TIME.
      NWL = NWP
      NPL = NPCELL/60
      DO 210 JX=1,NPL
      NWK = NWL + ( 60* ( NPLOTS+1 ) ) - 1
      READ (N2) (PLOT(I,1), I=NWL,NWK)
      NWL = NWK+1
 210  CONTINUE
      NPL = NPLOTS+1
      NWL = NWP
      DO 230 JX=1,NPNT
      TIME(JX) = PLOT(NWL,1)
      DO 220 I=1,NPLOTS
      NWL = NWL+1
      PLOT(JX,I) = PLOT(NWL,1)
 220  CONTINUE
      NWL = NWL+1
 230  CONTINUE
      CALL CLOCK (6HRSTIME,IDUM)
      DO 21 J=1, NPLOTS
      REF = PLOT(1,J)
      AMAX = REF
      AMIN = REF
```

```
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*(AMIN-REF)+REF
      CALL FINDER ( IMIN,NPNT,1,IMIN,T4,T1,PLOT(1,J),TIME )
      PT1 = .9*(AMIN-REF)+REF
      CALL FINDER ( IMIN,NPNT,1,IMIN,T3,T2,PLOT(1,J),TIME )
      GO TO 43
   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 ( 1,IMAX,IMAX,NPNT,T1,T4,PLOT(1,J),TIME )
      PT1 = .9*(AMAX-REF)+REF
      CALL FINDER ( 1,IMAX,IMAX,NPNT,T2,T3,PLOT(1,J),TIME )
   43 CONTINUE
C**** WRITE OUT RISE AND FALL TIMES.
C * * COMMENT BELOW SUPPRESSES PRINT OF RISE AND FALL TIMES.
C     WRITE (N6,50) TITLE(J),T1,T2,T3,T4
C * * *
   21 CONTINUE
C**** IF ONLY RESPONSE TIMES DESIRED, RETURN.
      IF (IPLOTI.EQ.4) GO TO 80
C
C             INSERT PLOTTING ROUTINES HERE.
      CALL PLTDTA(PLOT(NWP,1),NR,NC,TIME,PLOT,NPNT,NPLOTS,
     1  TITLE,DELT,NPNT)
C
      CALL CLOCK (6H*PLOT*,IDUM)
C * * NEXT TWO COMMENTS SUPPRESS LINE PRINTER PLOTS.
C     CALL PLOTER ( PLOT(NWP,1),NR,NC,TIME,PLOT,NPNT,NPLOTS,
C    1  TITLE,DELT,NPNT)
C    1 NPNT )
C * * *
      CALL CLOCK (6HR-PLOT,IDUM)
   80 CONTINUE
      RETURN
   30 FORMAT(////52X15HRESPONSE  TIMES//31X9HPARAMETER4X
     1 8H10  RISE 4X8H90  RISE4X8H10  FALL4X8H90  FALL/)
   50 FORMAT(33XA6,1X1P4E12.4)
      END
```

```
      SUBROUTINE PLTDTA(NSCR,NR,NC,TIME,PLOT,NPNT,NPLOTS,
     1  TITLE,DELT,NDIM)
      DIMENSION PLOT(NDIM,NPLOTS), TITLE(NPLOTS)
      DIMENSION NSCR(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.

```
      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(6,2001)
      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 50
      WRITE(6,102)
      READ(5,100) ISTART
      DO 10 I = 1,N
      ITQ = I + ISTART - 1 + (ISWTCH - 1) * NPNT
   10 A(I) = CMPLX(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
```

```
   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 CPLOTF(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. PDPRT) WRITE(6,203) TP
      PDPRT = .TRUE.
      WRITE(6,103)
      READ(5,100) JSWTCH
      IF(JSWTCH .LT. 100) GO TO 21
      CALL PRNT(A,N)
      JSWTCH = JSWTCH - 100
   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 * *
```

```
      JSWTCH = JSWTCH - 10
      IF(JSWTCH .EQ. 0) GO TO 1
   31 CONTINUE
      WRITE(6,304) NP
      WRITE(6,303)
      READ(5,100) NSTART
      NST = NSTART
      WRITE(6,305)
      READ(5,100) NSTOP
      NSP = NSTOP
      WRITE(6,306)
      READ(5,100) NJUMP
      CALL TTTP(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 FLO-)
  105 FORMAT(- ENTER FHI-)
  107 FORMAT(//- ERROR--LARGEST DATA SET NUMBER IS  -,I2/)
  108 FORMAT(- ENTER THE HIGHEST DESIRED FREQ IN THE SPECTRUM, FMAX-/)
  200 FORMAT(- THE NUMBER OF POINTS PER DATA SET = -,I5)
  201 FORMAT(- THE NUMBER OF DATA SETS = -,I2/)
  202 FORMAT(- THE NUMBER OF POINTS USED IN A TRANSFORM = -,I5)
  203 FORMAT(- THE PERIOD OF THE TIME FUNCTION = -,1PE11.4,- SEC-/)
  303 FORMAT(- ENTER NSTART-)
  304 FORMAT(- ARRAY SIZE = -,I5,-  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(0.,0.,999)
      STOP
      END
```

```
C * *
      SUBROUTINE CPLOTF(A,N,DELF,FMAX)
COMMENT-  THE FOLLOWING CONTROL STATEMENT MUST PRECEED THE
C         EXECUTE STATEMENT FOR RUNS USING CALCOMP PLOTS.
C
C     @USE UNIT #, TPFS
C * *
      COMPLEX A(1)
      IR = FMAX / DELF + .001
      FMAX = IR * DELF
      FLO = - FMAX
      FHI = FMAX
 1799 FORMAT()
      XTEST = ABS(FHI-FLO)
      IF(XTEST.LT.1.E-30) GO TO 9999
      XTEST = XTEST/(ABS(FLO) + ABS(FHI))
      IF(XTEST.LT.1.E-30) GO TO 9999
      YSPRED = 70.
      N2 = N/2
      NST = N2 - IR + 1
      NSP = N2 + IR + 1
      T1 = 1.E-35
      DO 4000 I = NST,NSP
      T2 = CABS(A(I))
 4000 IF(T2.GT.T1) T1 = T2
      DBMAX = 20.*ALOG10(T1)
      CALL SCALE(DBMAX,MAXSCL)
      XMXSCL = MAXSCL
      XKSCAL = 10.**(-XMXSCL/20.)
      CALL FACTOR(0.4)
      CALL PLOT(0.,-20.,3)
      CALL PLOT(12.,0.,-3)
      CALL PLOT(-10.,-14.,3)
      DO 3 I = 1,2
      CALL PLOT(-10.,0.,2)
      CALL PLOT(10.,0.,2)
      CALL PLOT(10.,-14.,2)
      CALL PLOT(-10.01,-14.,2)
      CALL PLOT(-10.01,0.01,2)
      CALL PLOT(10.01,0.01,2)
      CALL PLOT(10.01,-14.01,2)
    3 CALL PLOT(-10.,-14.01,2)
C     DO 3000 IAGAIN = 1,2
C * *
C      DO LOOP 10 CALIBRATES THE LEFT BORDER
C * *
      DO 10 I = 0,70
      Y = -14. + 0.2*I
      IF (MOD(I,5) .EQ. 0) GO TO 6
```

```
      CALL PLOT(-10.1,Y,3)
      GO TO 8
    6 IF (MOD(I,10) .EQ. 0) GO TO 7
      CALL PLOT(-10.16,Y,3)
      GO TO 8
    7 CALL PLOT(-10.2,Y,3)
    8 CALL PLOT(-10.,Y,2)
   10 CONTINUE
      Y = 2.
      DO 15 I = 1,8
      J = I - 1
      Y = Y-2.
      YY = XMXSCL - 10.*J*YSPRED/70.
   15 CALL NUMBER(-11.09,Y-.105,.21,YY,0.,-1)
      CALL SYMBOL(-11.24,-8.4,.21,14HAMPLITUDE (DB),90.0,14)
      FCENTR = (FLO + FHI)/2.
      FUPPER = FHI - FCENTR
      IF(FUPPER.GT.1.0) GO TO 2100
      IEXP = ALOG10(FUPPER) - 1
      GO TO 2101
 2100 CONTINUE
      XIEXP = ALOG10(FUPPER)
      IEXP = XIEXP
      RIEXP = IEXP
      IF((ABS(XIEXP-RIEXP).LT.1.E-20).AND.(XIEXP.GE.RIEXP))
     1   IEXP = IEXP - 1
 2101 CONTINUE
      FULSCL = FUPPER*(10.**(-IEXP))
      IFLSCL = FULSCL
      ITEMP = 10.*FULSCL
      RITEMP = ITEMP
      SCALE1 = IFLSCL
      SCALE1 = 10.*SCALE1/FULSCL
      TENIFS = 10.*IFLSCL
      SCALE1 = SCALE1/TENIFS
C * *
C       DO LOOP 40 CALIBRATES THE BOTTOM POSITIVE BORDER
C * *
      DO 40 I = 0,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
```

```
C * *
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 - -
C      DO LOOP 240 CALIBRATES THE TOP POSITIVE BORDER
C * *
      DO 240 I = 0,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,0.,2)
  240 CONTINUE
C * *
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,0.,2)
  340 CONTINUE
C - -
C        DO LOOP 20 CALIBRATES THE RIGHT HAND BORDER
C * *
      DO 20 I = 0,70
      Y = -14. + 0.2*I
      IF (MOD(I,5) .EQ. 0) GO TO 16
      CALL PLOT( 10.1,Y,3)
      GO TO 18
```

```
   16 IF (MOD(I,10) .EQ. 0) GO TO 17
      CALL PLOT( 10.16,Y,3)
      GO TO 18
   17 CALL PLOT( 10.2,Y,3)
   18 CALL PLOT( 10.,Y,2)
   20 CONTINUE
      AK1 = 10./FULSCL
      CALL NUMBER(- 0.06,-14.5,.21,     0.,0.,-1)
      DO 200 I = 1,IFLSCL
      XPOS = -.06 + I*AK1
      XNEG = -.12 - I*AK1
      CALL NUMBER(XPOS,-14.5,.21,1.*I,0.,-1)
  200 CALL NUMBER(XNEG,-14.5,.21,-1.*I,0.,-1)
      CALL SYMBOL(-4.5,-14.9,.21,
     1 46H(FREQUENCY - FCENTER) DIVIDED BY FSCALE,  (HZ),0.,46)
      CALL SYMBOL(-4.5,-15.3,.21,10HFCENTER = ,0.,10)
      CALL NUMBER(-2.0,-15.3,.21,FCENTR,0.,0)
      CALL SYMBOL(0.,-15.3,.21,9HFSCALE = ,0.,9)
      CALL NUMBER(2.4,-15.3,.21,10.**IEXP,0.,0)
C3000 CONTINUE
      ISTOP = NSP - NST + 1
      DENOM = NSP - NST
      DELX = 20./DENOM
      XS = - (N2 + 1 - NST) * DELX
      DO 30 I = 1,ISTOP
      II = I - 1
      T1 = (CABS(A(I+NST-1)))*XKSCAL
      IF(T1 .LT. 1.E-7) GO TO 30
      Y1 = (70./YSPRED) * 4. * ALOG10(T1)
      IF(Y1 .LE. -14.0) GO TO 30
      X1 = II * DELX + XS
      IF(X1 .LT. -10.) GO TO 30
      IF(X1 .GT. 10.) GO TO 30
      IF(Y1.LE.0.) GO TO 250
      CALL SYMBOL(X1,.16,.21,1H*,0.,1)
      GO TO 30
  250 CONTINUE
      CALL PLOT(X1,-14.,3)
      CALL PLOT(X1,Y1,2)
      CALL PLOT(X1,-14.,2)
   30 CONTINUE
      CALL PLOT(13.,-20.,-3)
   35 FORMAT(1H1,2X,14HPLOT COMPLETED)
      WRITE(6,35)
 9999 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FFT(A,IGAM,ISN)
      COMPLEX A(1),T1,T2,TEMP
      DOUBLE PRECISION PI2,SO,CO,SI,CI,SN,CS
      PI2 = 6.2831853071795864BD0
      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 .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 = 0.0D0
      CO = 1.0D0
      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(1)
      WRITE(6,101)
      DO 10 I = 1,N
      DB = 1.E30
      T = CABS(A(I))
      IF(T .GT. 0.) DB = 20. * ALOG10(T)
   10 WRITE(6,100) I,A(I),DB
  100 FORMAT(1X,I5,1P2E15.4,5X,0PF8.2)
  101 FORMAT(//,-  LINE-,7X,-REAL-,11X,-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(DBMAX,MAXSCL)
C * * *  THIS SUBROUTINE ESTABLISHES ORDINATE SCALING FOR
C        THE REMOTE SPECTRUM PLOTTER.
      IF(DBMAX.LE.0.) GO TO 10
      MAXSCL = 0
    1 MAXSCL = MAXSCL + 10
      DIFF = DBMAX - MAXSCL
      IF(DIFF.GT.0.) GO TO 1
      GO TO 999
   10 MAXSCL = 0
   11 MAXSCL = MAXSCL - 10
      DIFF = (DBMAX - MAXSCL)
      IF(DIFF.LE.0.) GO TO 11
      MAXSCL = MAXSCL + 10
  999 RETURN
      END
```

```
      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
      DECTMP = 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 * * * *
C * * * *
      WRITE(6,2) IEND
    2 FORMAT(/5X,8HNSIZE = ,I5/)
C * * * *
      CALL SCALE(DBMAX,MAXSCL)
C * * * *
C    THE ORDINATE WILL VARY FROM(MAXSCL-50) DB UP
C    TO MAXSCL DB.
      DO 33 I = 1,50
   33 IA(I) = 1H
      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)
```

```
 200 FORMAT(2X,14HFREQUENCY (HZ),9X,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,I4,4(6X,I4),5X,I4)
     WRITE(6,8)
   8 FORMAT(9X,1HI,5(10H----+----I))
     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 + I - 1))
     IF(DECTMP.LT.1.E-30) DECTMP = 1.E-30
     B = 20. * ALOG10(DECTMP)
     M = 50 + B - MAXSCL
     J = I - 1
     XJ = J
     FREQ = FLOPRT + XJ * DELF1
     IF(M.LT.0) GO TO 50
     IF(M.EQ.0) GO TO 34
     DO 1515 II = 1,M
1515 IA(II) = 1H-
     WRITE(6,35) FREQ,IA
     DO 1616 II = 1,M
1616 IA(II) = 1H
  35 FORMAT(F8.3,2H I,50A1)
     GO TO 36
  34 WRITE(6,37) FREQ
  37 FORMAT(1X,F7.3,2H -)
     GO TO 36
  50 WRITE(6,51) FREQ
  51 FORMAT(1X,F7.3,2H I)
1000 FORMAT( )
  36 CONTINUE
  10 CONTINUE
     WRITE(6,11)
  11 FORMAT(6X,4HFREQ)
     RETURN
     END
```

```
      SUBROUTINE TTTP(A,NST,NSP,NJUMP)
      COMPLEX A(1)
      DIMENSION IA(50)
      BMAX = 0.
      BMIN = BMAX
      DO 1 I = NST,NSP,NJUMP
      B = REAL(A(I))
      IF(B.LT.BMIN) BMIN = B
    1 IF(B.GT.BMAX) BMAX = B
      IF((BMAX-BMIN).LT.1.E-30) GO TO 999
      DO 33 I = 1,50
   33 IA(I) = 1H
  100 WRITE(6,4)
    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(8X,2H 0,3X,2H.1,3X,2H.2,3X,2H.3,3X,2H.4,3X,2H.5,
     1    3X,2H.6,3X,2H.7,3X,2H.8,3X,2H.9,2X,3H1.0)
      WRITE(6,7)
    7 FORMAT(1H,8X,1HI,10(5H----I))
      DO 10 I = NST,NSP,NJUMP
      B = REAL(A(I))
      B = (B-BMIN)/(BMAX-BMIN)
      M = INT(B*50.+0.5)
      IF(M.EQ.0) GO TO 34
      IA(M) = 1H*
      WRITE(6,35) I,IA
      IA(M) = 1H
   35 FORMAT(2X,I5,3H  I,50A1)
      GO TO 36
   34 WRITE(6,37) I
   37 FORMAT(2X,I5,3H  *)
   36 CONTINUE
   10 CONTINUE
      WRITE(6,11)
   11 FORMAT(6X,1HN)
      GO TO 900
  999 WRITE (6,9)
    9 FORMAT(1X,12HERROR 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)
      WRITE(6,100) 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(30X,-TIME-/)
  112 FORMAT(25X,-DATA SET NUMBER  -,I2/)
  113 FORMAT(//- ERROR--LARGEST DATA SET NUMBER IS  -,I2/)
  999 WRITE(6,102)
      RETURN
      END
```

APPENDIX C

FATCAT PROGRAM DESCRIPTION AND LISTING

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 CAL-COMP 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 C 1 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.

TABLE C 1

ALPHABETICAL LISTING OF ALL SUBROUTINE ENTRIES IN FATCAT

(All names denote subroutines unless marked otherwise)

1. ADJN
2. AMDEMO
3. AMP
4. BCDFPT
5. BWBNDP
6. BWBSTP* (BWBNDP)
7. BWHIP* (BWBNDP)
8. BWLOWP* (BWBNDP)
9. CHBNDP
10. CHBSTP* (CHBNDP)
11. CHHIP* (CHBNDP)
12. CHLOWP* (CHBNDP
13. CPLOTF**
14. ELFIND
15. FACTOR+
16. FETCH
17. FFT
18. FILTER
19. FLATSP
20. FMDEMO
21. FRQFCN* (TIMFCN)
22. FRQMUL
23. IDLMUL
24. INPFOR
25. LFOLD
26. LIM
27. LSTCOM
28. NUMBER+
29. PDCHK
30. PERIOD
31. PHDEMO
32. PLOT+
33. PLOTS+
34. PROCES
35. PRTFAC* (PERIOD)
36. SCALE
37. SIGGEN
38. STRDTA
39. SYMBOL+
40. SYNBP
41. SYNHP* (SYNRP)
42. SYNLP* (SYNBP)
43. TELPLT
44. TIMFCN
45. TTFP
46. WRFF* (WRTF)
47. WRTF

*Entry Point in subroutine named in parenthesis.

**Not used in SIGMA-5 version.

+CALCOMP plotter subroutines called by CPLOT.

2. MAIN PROGRAM

Calls: PLOTS*, FETCH, ELFIND, WRTF, WRFF, TELPLT, TTFP, CPLOTF, PRTFAC, STRDTA, PROCES, LSTCOM, INPFOR, PDCHK.

Commons: blank, CFREQ, CDOM, CDATA, CCIRKT, 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,CARRFQ
      COMMON /CFREQ/ NFR,FR(6)
      COMMON /CDOM/ DOMFLG
      COMMON /CDATA/ JCTR,DATA(200)
      COMMON /CCIRKT/ NBLK,ITYP(30,2)
      COMMON/CWORD/ WORD(10)
      COMMON/CFLGS/ PDFLG,ARFLG
      LOGICAL PDFLG, ARFLG
C     DIMENSION IBUF(5000)
C     CALL PLOTS(IBUF(1),5000,2)
      JCTR = 1
      ITYP(1,2) = JCTR
      NFR = 0
      NBLK = 0
      IBLK = 0
      PDFLG = .FALSE.
      ARFLG = .FALSE.
      WRITE(6,7006)
    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,
     1       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
```

*CALCOMP plotter routine.

```
   10 IF (WORD(3) .EQ. 1H ) GO TO 12
      N1 = WORD(2)
      N2 = WORD(3)
      GO TO 15
   12 WRITE(6,7004)
      READ(5,7000) N1,N2
   15 CALL WRTF(A,N1,N2)
      GO TO 1
   20 IF (WORD(3) .EQ. 1H ) GO TO 22
      FRLO = WORD(2)
      FRHI = WORD(3)
      GO TO 25
   22 WRITE(6,7005)
      READ(5,7000) 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
      NST = WORD(2)
      NSP = WORD(3)
      NJUMP = WORD(4)
      IF (NJUMP .EQ. 1H ) NJUMP = 1
      GO TO 35
   32 WRITE(6,7007)
      READ(5,7000) 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,2(I3,3X))
      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 STRDTA(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 STRDTA(2,1,2)
      NTYP = 16
      GO TO 600
  240 GO TO 430
C * * FM DEMODULATOR
  250 CALL STRDTA(1,0,0)
      NTYP = 19
      GO TO 600
C * * PHASE DEMODULATOR
  260 CALL STRDTA(1,0,0)
      NTYP = 20
      GO TO 600
C * * AMPLIFIER
  270 CALL STRDTA(1,0,0)
      NTYP = 2
      GO TO 600
C * * LIMITER
  280 CALL STRDTA(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)
  292 CONTINUE
      IBLK = NOUT
  295 WRITE(6,7002) IBLK
      GO TO 1
```

```
  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 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)
      GO TO 1
 7000 FORMAT()
 7001 FORMAT(- * * ERROR * * LARGEST BLOCK NO IS -,I2,- * *-)
 7002 FORMAT(- PROCESSING COMPLETE THRU BLOCK -,I2)
 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 -,A6,- IS NOT YET OPERATIONAL-)
  999 CONTINUE
C     CALL PLOTS(0.,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 wave form; conversion to a real time wave form 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,CARRFQ
      COMMON/CDATA/ JCTR,DATA(200)
      COMPLEX A(1)
      PI2 = 6.2831853
      N2 = N /2
      FO = 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
      IFO = FO / DELF + .5
      NSTART = N2 + 1
      NSTOP1 = N - IFO + 1
      DO 11 I = NSTART,N
   11 A(I - IFO) = A(I)
      DO 12 I = NSTOP1,N
   12 A(I) = (0., 0.)
C * * RECOVER THE AMPLITUDE INFORMATION
      CALL TIMFCN(A)
      DO 20 I = 1,N
      TEMP = CABS(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(1)
      COMMON N,IGAM
      COMMON/CDATA/ JCTR,DATA(200)
      XP = DATA(JCTR) / 20.
      G = 10.**XP
      DO 100 I = 1,N
  100 A(I) = G*A(I)
      RETURN
      END
```

6. Function BCDFPT

Called by: FETCH

Calls: none

Commons: none

Entries: none

Description: BCDFPT 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 BCDFPT( BCD,N )
C          BCDFPT CONVERTS DATA FROM BCD TO FLOATING POINT.
C          BCD IS AN ARRAY CONTAINING THE N BCD CHARACTERS
C          WHICH ARE TO BE CONVERTED.
C             I = INDEX OF THE CHARACTER BEING CONVERTED.
C             J = INDEX CORRESPONDING TO THE DIGIT J-1.
C             K = 1 WHEN DECODING WHOLE NUMBER PORTION.
C                 2 WHEN DECODING FRACTIONAL PORTION.
C                 3 WHEN DECODING EXPONENT.
      INTEGER DIGIT, E, PLUS, DECPT, BCD
      LOGICAL EXPFLG, DIGFLG, DECFLG, EXSIGN
      DIMENSION BCD(1),KSIGN(3),INTEGR(3),RESULT(3),DIGIT(10)
      DATA DIGIT / 1H0,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9/
      DATA PLUS, MINUS, E, DECPT / 1H+, 1H-, 1HE, 1H. /
      EXPFLG = .FALSE.
      DIGFLG = .FALSE.
      DECFLG = .FALSE.
      EXSIGN = .FALSE.
      DO 11 K=1,3
      KSIGN(K) = 1
      INTEGR(K) = 0
   11 CONTINUE
      NPLART = 0
      K = 1
      DO 31 I=1,N
      ICHAR = BCD(I)
```

```
C*
C         TEST FOR SIGN, DIGIT, DECIMAL POINT, OR E
C*
      IF ( ICHAR-PLUS ) 13,23,13
   13 IF ( ICHAR-MINUS ) 14,24,14
   14 DO 15 J=1,10
      IF ( ICHAR-DIGIT(J) ) 15,25,15
   15 CONTINUE
      IF ( ICHAR-DECPT ) 16,26,16
   16 IF ( ICHAR-E ) 21,29,21
C*
C*                                              PLUS SIGN
C*
   23 IF ( DIGFLG ) GO TO 28
      GO TO 31
C*
C*                                              MINUS SIGN
C*
   24 IF ( DIGFLG ) GO TO 27
      KSIGN(1) = -1
      GO TO 31
C*
C*                                              DIGIT FROM 0 TO 9
C*
   25 INTEGR(K) = 10*INTEGR(K)+J-1
      NPLART = NPLART+K-1
      DIGFLG = .TRUE.
      GO TO 31
C*
C*                                              DECIMAL POINT
C*    ONLY ONE DECIMAL POINT PER NUMBER IS ALLOWED.
C*    DECIMAL POINT IS NOT ALLOWED IN EXPONENT.
C*
   26 IF ( DECFLG ) GO TO 21
      IF ( EXPFLG ) GO TO 21
      DECFLG = .TRUE.
      K = 2
      GO TO 31
```

```
C*
C*                                                *E* FOR EXPONENT
C*     BLANK TIMES TEN ** EXPONENT NOT ALLOWED.
C*
   27 KSIGN(3) = -1
   28 IF ( EXSIGN ) GO TO 21
      EXSIGN = .TRUE.
      GO TO 30
   29 IF ( EXPFLG ) GO TO 21
      IF ( .NOT. DIGFLG ) GO TO 21
   30 EXPFLG = .TRUE.
      K = 3
      NPLASV = NPLART
   31 CONTINUE
C*
C           THE NUMBER HAS BEEN SEPARATED INTO INTEGER, FRACTION,
C           AND EXPONENT PARTS.  COMBINE THEM TO FORM THE
C           NUMBER IN FLOATING POINT.
C
      IF ( EXPFLG ) GO TO 32
      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 = 10.**NPLART
      RESULT(1) = FLOAT( INTEGR(1)*KSIGN(1) )
      RESULT(2) = FLOAT( INTEGR(2)*KSIGN(1) ) / RTSHFT
      BCDFPT = ( RESULT(1)+RESULT(2) )*EXPON
   41 RETURN
C*
C           ILLEGAL 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, BWLOWP, 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 maximmally flat response defined by the function

$$\left| H\,(\omega_p) \right| = \frac{1}{\sqrt{1 + (\omega_p{}^2)^n}} \tag{1}$$

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

$$S_k = e^{j\theta} \tag{2}$$

where

$$\theta = \left(\frac{2k + n - 1}{n}\right)\left(\frac{\pi}{2}\right), \; k = 1, 2, \ldots 2n. \tag{3}$$

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

$$H(S) = \frac{1}{(S - S_1)(S - S_2)(S - S_3) \ldots (S - S_n)} \tag{4}$$

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, ω_p, can be determined to satisfy the definitions

$$\omega_p = \omega/\omega_c \text{ for low pass,}$$

$$\omega_p = \frac{\omega - \omega_o}{\left(\frac{B\omega}{2}\right)} \text{ for band pass,}$$

$\omega_p = \omega_c/\omega$ for high pass,

$$\omega_p = \frac{\frac{B\omega}{2}}{\omega - \omega_o} \text{ for band stop,}$$

where

ω_c = 3 dB corner frequency, and

$B\omega$ = 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:

```
      SUBROUTINE BWBNDP(A)
      COMPLEX A(1),S(20)
      COMMON/CDATA/ JCTR,DATA(200)
      COMMON/CFILT/ FO,FCOFF,NR, AMP,FFLG,S
      LOGICAL FFLG
      FFLG = .TRUE.
      GO TO 2
      ENTRY BWBSTP(A)
      FFLG = .FALSE.
    2 FO = DATA(JCTR)
      FCOFF = DATA(JCTR+1) / 2.
      NR = DATA(JCTR+2)
      GO TO 6
      ENTRY BWLOWP(A)
      FFLG = .TRUE.
      GO TO 4
      ENTRY BWHIP(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

$$S_k = \sigma_k + j\omega_k \qquad (1)$$

where

$$\sigma_k = \pm \tanh a \sin \theta ,$$

$$\omega_k = \cos \theta$$

$$a = \frac{1}{n} \sinh^{-1} \frac{1}{\epsilon} ,$$

$$\theta = \left(\frac{2k - 1}{n}\right)\left(\frac{\pi}{2}\right), \; k = 1, 2, 3, \ldots 2n,$$

$$\epsilon = \text{ripple width}, \; 0 < \epsilon < 1.$$

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 F0 = DATA(JCTR)
      FCOFF = DATA(JCTR+1) / 2.
      NR = DATA(JCTR+2)
      EPSDB = DATA(JCTR+3)
      GO TO 6
      ENTRY CHLOWP(A)
      FFLG = .TRUE.
      GO TO 4
      ENTRY CHHIP(A)
      FFLG = .FALSE.
    4 F0 = 0.
      FCOFF = DATA(JCTR)
      NR = DATA(JCTR+1)
      EPSDB = DATA(JCTR+2)
    6 X = 1. / SQRT(EXP(.23025851 * EPSDB) - 1.)
      ARG = X + SQRT(X ** 2 + 1)
      AE = ALOG(ARG) / NR
      CALL FRQFCN(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 * CABS(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 embodys 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)
COMMENT-   THE FOLLOWING CONTROL STATEMENT MUST PRECEED THE
C          EXECUTE STATEMENT FOR RUNS USING CALCOMP PLOTS.
C
C     @USE UNIT #, TPF$
C * *
      COMMON N,IGAM,DELF
      COMPLEX A(1)
      CALL FRQFCN(A)
 1799 FORMAT()
      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-30) GO TO 9999
      XTEST = XTEST/(ABS(FLO) + ABS(FHI))
      IF(XTEST.LT.1.E-30) GO TO 9999
      YSPRED = 70.
      N2 = N/2
      NST = N2 + INT(FLO/DELF) + 1
      NSP = N2 + INT(FHI/DELF) - 1
      T1 = 1.E-35
      DO 4000 I = NST,NSP
      T2 = CABS(A(I))
 4000 IF(T2.GT.T1) T1 = T2
      DBMAX = 20.*ALOG10(T1)
      CALL SCALE(DBMAX,MAXSCL)
      XMXSCL = MAXSCL
      XKSCAL = 10.**(-XMXSCL/20.)
```

*CALCOMP plotter routines

```
      CALL FACTOR(0.4)
      CALL PLOT(0.,-20.,3)
      CALL PLOT(12.,0.,-3)
      CALL PLOT(-10.,-14.,3)
      DO 3 I = 1,2
      CALL PLOT(-10.,0.,2)
      CALL PLOT(10.,0.,2)
      CALL PLOT(10.,-14.,2)
      CALL PLOT(-10.01,-14.,2)
      CALL PLOT(-10.01,0.01,2)
      CALL PLOT(10.01,0.01,2)
      CALL PLOT(10.01,-14.01,2)
    3 CALL PLOT(-10.,-14.01,2)
      DO 3000 IAGAIN = 1,2
      DO 10 I = 0,70
      Y = -14. + 0.2*I
      IF (MOD(I,5) .EQ. 0) GO TO 6
      CALL PLOT(-10.1,Y,3)
      GO TO 8
    6 IF (MOD(I,10) .EQ. 0) GO TO 7
      CALL PLOT(-10.16,Y,3)
      GO TO 8
    7 CALL PLOT(-10.2,Y,3)
    8 CALL PLOT(-10.,Y,2)
   10 CONTINUE
      Y = 2.
      DO 15 I = 1,8
      J = I - 1
      Y = Y-2.
      YY = XMXSCL - 10.*J*YSPRED/70.
   15 CALL NUMBER(-11.09,Y-.105,.21,YY,0.,-1)
      CALL SYMBOL(-11.24,-8.4,.21,14HAMPLITUDE (DB),90.0,14)
      FCENTR = (FLO + FHI)/2.
      FUPPER = FHI - FCENTR
      IF(FUPPER.GT.1.0) GO TO 2100
      IEXP = ALOG10(FUPPER) - 1
      GO TO 2101
 2100 CONTINUE
      XIEXP = ALOG10(FUPPER)
      IEXP = XIEXP
      RIEXP = IEXP
      IF((ABS(XIEXP-RIEXP).LT.1.E-20).AND.(XIEXP.GE.RIEXP))
     1   IEXP = IEXP - 1
 2101 CONTINUE
      FULSCL = FUPPER*(10.**(-IEXP))
      IFLSCL = FULSCL
      ITEMP = 10.*FULSCL
      RITEMP = ITEMP
      SCALE1 = IFLSCL
```

```
      SCALE1 = 10.*SCALE1/FULSCL
      TENIFS = 10.*IFLSCL
      SCALE1 = SCALE1/TENIFS
      DO 40 I = 0,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
      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
      DO 240 I = 0,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,0.,2)
  240 CONTINUE
      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,0.,2)
  340 CONTINUE
```

```
      DO 20 I = 0,70
      Y = -14. + 0.2*I
      IF (MOD(I,5) .EQ. 0) GO TO 16
      CALL PLOT( 10.1,Y,3)
      GO TO 18
   16 IF (MOD(I,10) .EQ. 0) GO TO 17
      CALL PLOT( 10.16,Y,3)
      GO TO 18
   17 CALL PLOT( 10.2,Y,3)
   18 CALL PLOT( 10.,Y,2)
   20 CONTINUE
      AK1 = 10./FULSCL
      CALL NUMBER(- 0.06,-14.5,.21,       0.,0.,-1)
      DO 200 I = 1,IFLSCL
      XPOS = -.06 + I*AK1
      XNEG = -.12 - I*AK1
      CALL NUMBER(XPOS,-14.5,.21,1.*I,0.,-1)
  200 CALL NUMBER(XNEG,-14.5,.21,-1.*I,0.,-1)
      CALL SYMBOL(-4.5,-14.9,.21,
     1 46H(FREQUENCY - FCENTER) DIVIDED BY FSCALE,    (HZ),0.,46)
      CALL SYMBOL(-4.5,-15.3,.21,10HFCENTER = ,0.,10)
      CALL NUMBER(-2.0,-15.3,.21,FCENTR,0.,0)
      CALL SYMBOL(0.,-15.3,.21,9HFSCALE = ,0.,9)
      CALL NUMBER(2.4,-15.3,.21,10.**IFXP,0.,0)
 3000 CONTINUE
      ISTOP = NSP - NST + 1
      DENOM = NSP - NST + 2
      DELX = 20./DENOM
      DO 30 I = 1,ISTOP
      T1 = (CABS(A(I+NST)))*XKSCAL
      IF(T1 .LT. 1.E-7) GO TO 30
      Y1 = (70./YSPRED) * 4. * ALOG10(T1)
      IF(Y1 .LE. -14.0) GO TO 30
      X1 = I * DELX - 10.
      IF(X1 .LT. -10.) GO TO 30
      IF(X1 .GT. 10.) GO TO 30
      IF(Y1.LE.0.) GO TO 250
      CALL SYMBOL(X1,.16,.21,1H*,0.,1)
      GO TO 30
  250 CONTINUE
      CALL PLOT(X1,-14.,3)
      CALL PLOT(X1,Y1,2)
      CALL PLOT(X1,-14.,2)
   30 CONTINUE
      CALL PLOT(13.,-20.,-3)
   35 FORMAT(1H1,2X,14HPLOT COMPLETED)
      WRITE(6,35)
 9999 CONTINUE
      RETURN
      END
```

10. Subroutine ELFIND

Called by: INPFOR, MAIN, PDCHK

Calls: none

Commons: none

Entries: none

Description: ELFIND 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 ELFIND (NAME,L)
      PARAMETER NMAX = 42
C
C         ELFIND TRIES TO MATCH THE KEY WORD FROM AN INPUT DATA
C         STRING (NAME) AGAINST ONE OF THE ALLOWABLE INPUT FORMS.
C         L IS SET TO THE INDEX WHICH CORRESPONDS TO THE MATCHED
C         INPUT TYPE.
      DIMENSION MATCH(NMAX)
      DATA (MATCH(I), I=1,NMAX)
     A / 6HPRINTT,6HPRINTF,6HTPLOTT,6HTPLOTF,6HCPLOTT,6HCPLOTF,
     B   6HPRIMEF,6HENDOFJ,6HBWBNDP,6HBWLOWP,5HBWHIP ,6HBWBSTP,
     C   6HCHBNDP,6HCHLOWP,5HCHHIP, 6HCHBSTP,5HSYNBP, 5HSYNLP,
     D   5HSYNHP, 6HSIGGEN,6HFROMUL,6H      ,6HIDLMUL,6H      ,
     E   6HFMDEMO,6HPHDEMO,3HAMP,   3HLIM,   5HBLOCK, 3HYES,
     F   2HNO,    1HN,     6H      ,6HAMDEMO,6HFLATSP,6HLISTCO,
     G   6HCIRCUI,6HINPUTF,6HDELETE,6HINSERT,6HREPLAC,6HREPEAT/
      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: BCDFPT

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 BCDPFT 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(WORD,LL,NBAD)
      INTEGER APOST,BLANK,COMMA,DECPT,EQUAL,PLUS,RPAREN,TEST
      INTEGER BUFF1,BUFF2,BUFF3,BCDFPT,TITLE,WORD,E
      DIMENSION TITLE(12)
      COMMON/CFETCH/ BUFF2(6),BUFF1(80)
      DIMENSION WORD(1)
      DATA APOST,BLANK,COMMA,DECPT,EQUAL,MINUS,NINE,NZ,PLUS
     1     / 1H- ,1H  ,1H,,1H. ,1H= ,1H- ,1H9 ,1H0 ,1H+ /
      DATA LPAREN,RPAREN
     1     /  1H( ,  1H) /
      DATA E / 1HE /
      L=0
      NCOLS = 80
      NBAD = 1
C
C        FETCH IS A FREE-FIELD INPUT SUBPROGRAM WHICH RETURNS
C        THE INPUT DATA IN LL CONSECUTIVE CELLS OF THE
C        ARRAY WORD. HOLLERITH IS TRUNCATED TO 6 CHAR..
    1 CONTINUE
      READ  (5,1001,END = 107)   ( BUFF1(I),I=1,80 )
      WRITE (6,1005) ( BUFF1(I),I=1,80 )
   20 M = 0
    2 K = 0
      N = 0
      NCOMMA = 0
```

```
      DO 3 I = 1,6
    3 BUFF2(I) = BLANK
C
    4 IF ( M-NCOLS ) 40,100,100
C   *     EXAMINE EACH COLUMN, REMOVE BLANKS, AND TEST FOR
C         SEPARATORS.
   40 M = M+1
      TEST = BUFF1(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 ) 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.
    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 ) 55,30,55
   55 IF ( TEST - DECPT ) 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 BUFF3 = BCDFPT( BUFF2,N )
       IF ( N ) 106,106,8
     8 WORD(L+1) = BUFF3
       GO TO 91
     9 ENCODE(6,1002,WORD(L+1)) (BUFF2(I),I=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.
       IF ( M-NCOLS ) 2,10,10
    10 IF ( TEST - COMMA ) 11,1,11
    11 LL = L
       RETURN
C    *      MOVE A TITLE CARD INTO THE TITLE ARRAY.
    30 ENCODE(72,1001,TITLE) (BUFF1(I),I = 1,72)
       GO TO 1
C    *      A CARD IMAGE HAS BEEN PROCESSED.  IF THE LAST
C           NON-BLANK SYMBOL WAS A COMMA (NCOMMA=0), READ
C           THE NEXT CARD. OTHERWISE THERE IS INFORMATION
C           IN *BUFF2* TO BE CONVERTED, AFTER WHICH, FETCH
C           WILL RETURN TO THE CALLING PROGRAM.
   100 IF ( NCOMMA ) 6,1,6
C    *      FETCH FOUND CONCOMITANT SEPARATORS OR A NUMBER WITH
C           MORE THAN 15 DIGITS AND COULD NOT CONTINUE.
   106 WRITE (6,2000)  (BUFF2(I),I=1,6)
       NBAD = 0
       RETURN
   107 STOP
  1001 FORMAT(80A1)
  1002 FORMAT(6A1)
  1005 FORMAT(1X80A1)
  2000 FORMAT(25H0**  FETCH CANNOT DECODE 6A1,4H  **//)
       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.

*Walsh, J. R. and R. D. Wetherington, CCS Down-Link Spectral Studies, Technical Report No. 7, Contract NAS8-20054, Georgia Institute of Technology, 29 May 1970.

Program Listing:

```
      SUBROUTINE FFT(A,IGAM,ISN)
      COMPLEX A(1),T1,T2,TEMP
      DOUBLE PRECISION PI2,SO,CO,SI,CI,SN,CS
      PI2 = 6.2831853071795864800
      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 .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 = 0.0D0
      CO = 1.0D0
      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

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

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

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

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

Program Listing:

```
      SUBROUTINE FILTER(A)
      COMPLEX A(1),S(20),HD,Z
      COMMON N,IGAM,DELF,DELT
      COMMON/CFILT/ F0,FCOFF,NR,AMP,FFLG,S
      LOGICAL FFLG
      TEST = EXP(35. / NR)
      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 (FFLG) 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,NR
   20 HD = HD * (Z - S(K))
      A(I) = AMP * A(I) / HD
      GO TO 30
   25 A(I) = CMPLX(0.,0.)
   30 CONTINUE
      RETURN
      END
```

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 examing the transfer functions of filters in detail.

Program Listing:

```
      SUBROUTINE FLATSP(A)
      COMPLEX A(1)
      COMMON N,IGAM,DELF,DELT,PD
      COMMON /CDATA/ JCTR,DATA(200)
      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 2π). A tracking loop corrects for excursions beyond the $\pm\pi$ range thus reconstructing the phase deviation due to the angle modulation.

Program Listing:

```
      SUBROUTINE FMDEMO(A)
      COMMON N,IGAM,DELF,DELT,PD,CARRFQ
      COMMON/CDATA/ JCTR,DATA(200)
      COMPLEX A(1)
      PI2 = 6.2831853
      J = 0.
      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
      IF0 = F0 / DELF + .5
      NSTART = N2 + 1
      NSTOP1 = N - IF0 + 1
      DO 11 I = NSTART,N
   11 A(I - IF0) = A(I)
      DO 12 I = NSTOP1,N
   12 A(I) = (0., 0.)
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, 0.)
      A(1) = CMPLX(AIMAG(A(1)), 0.)
C * * ZERO THE D-C COMPONENT
      CALL FRQFCN(A)
      A(N2 + 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
      NDBL = 2 * N - 1
      DO 100 I = 1,NDBL,2
      A(I) = A(I) - 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(200)
      COMMON/CFREQ/ NFR,FR(6)
      COMPLEX A(1)
      AMPLO = DATA(JCTR)
      FLO = DATA(JCTR + 1)
      CALL TIMFCN(A)
      PI2 = 6.2831853
      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(10)
      LOGICAL FLG
      FLG = .TRUE.
      CALL ELFIND(WORD(2),L)
      GO TO (    1,    1,    1,    1,    1,    1,    1,    1,    9,   10,
     A          11,   12,   13,   14,   15,   16,   17,   18,   19,   20,
     B          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(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(6,7101)
      IF (FLG) GO TO 999
   10 WRITE(6,7010)
      WRITE(6,7102)
      IF (FLG) GO TO 999
   11 WRITE(6,7011)
      WRITE(6,7102)
      IF (FLG) GO TO 999
   12 WRITE(6,7012)
      WRITE(6,7101)
      IF (FLG) GO TO 999
   13 WRITE(6,7013)
      WRITE(6,7101)
      WRITE(6,7103)
      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
   17 WRITE(6,7017)
      WRITE(6,7101)
      IF (FLG) GO TO 999
   18 WRITE(6,7018)
      WRITE(6,7102)
      IF (FLG) GO TO 999
   19 WRITE(6,7019)
      WRITE(6,7102)
      IF (FLG) GO TO 999
   25 WRITE(6,7025)
      WRITE(6,7107)
      IF (FLG) GO TO 999
   34 WRITE(6,7034)
      WRITE(6,7107)
      IF (FLG) GO TO 999
   26 WRITE(6,7026)
      WRITE(6,7107)
      IF (FLG) GO TO 999
   21 WRITE(6,7021)
      WRITE(6,7105)
      IF (FLG) GO TO 999
   23 WRITE(6,7023)
      WRITE(6,7106)
      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(2) .EQ. 3HALL) GO TO 510
      WRITE(6,7112) WORD(2)
      GO TO 999
  510 FLG = .FALSE.
      GO TO 20
```

```
7000 FORMAT()
7001 FORMAT(1X,A6,- IS NOT AN INPUT COMMAND-)
7009 FORMAT(/- BWBNDP, FO, BW, NR-)
7010 FORMAT(/- BWLOWP, FC, NR-)
7011 FORMAT(/- BWHIP, FC, NR-)
7012 FORMAT(/- BWBSTP, FO, BW, NR-)
7013 FORMAT(/- CHBNDP, FO, BW, NR, EPSDB-)
7014 FORMAT(/- CHLOWP, FC, NR, EPSDB-)
7015 FORMAT(/- CHHIP, FC, NR, EPSDB-)
7016 FORMAT(/- CHBSTP, FO, BW, NR, EPSDB-)
7017 FORMAT(/- SYNBP, FO, BW, NR-)
7018 FORMAT(/- SYNLP, FC, NR-)
7019 FORMAT(/- SYNHP, FC, NR-)
7020 FORMAT(/- SIGGEN, FO, FMOD, AM, PM, FM, A-)
7021 FORMAT(/- FRQMUL-)
7023 FORMAT(/- IDLMUL, ALO, FLO-)
7025 FORMAT(/- FMDEMO, FO-)
7026 FORMAT(/- PHDEMO, FO-)
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(- FO = 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(- FO = CARRIER FREQ, HZ-/
    A        - FMOD = MODULATION FREQ, HZ-/
    B        - 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(- FO = 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
```

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)=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(1)
      CALL TIMFCN(A)
      DO 1 I = 1,N
      A(I) = GL*A(I)
      IF (REAL(A(I)) .LE. CLEVL) A(I) = CMPLX(CLEVL,0.)
    1 IF (REAL(A(I)) .GE. CLEVH) A(I) = CMPLX(CLEVH,0.)
      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 /CWORD/ WORD(10)
      WRITE(6,7001)
      I = 1
      IF (WORD(2) .EQ. 3HALL) GO TO 20
      I = 0
      IF (WORD(2) .EQ. 6HSOURCE) GO TO 20
      IF (WORD(2) .EQ. 6HFILTER) GO TO 30
      IF (WORD(2) .EQ. 6HDEMODU) GO TO 40
      IF (WORD(2) .EQ. 4HMISC) GO TO 50
      IF (WORD(2) .EQ. 6HCONTRO) GO TO 60
      IF (WORD(2) .EQ. 5HNSIZE) GO TO 70
      IF (WORD(2) .EQ. 6HLISTCO) GO TO 80
      WRITE(6,7020) WORD(2)
      GO TO 999
   20 WRITE(6,7002)
      IF (I .EQ. 0) GO TO 999
   30 WRITE(6,7003)
      IF (I .EQ. 0) GO TO 999
   40 WRITE(6,7004)
      IF (I .EQ. 0) GO TO 999
   50 WRITE(6,7005)
      IF (I .EQ. 0) GO TO 999
   60 WRITE(6,7006)
      IF (I .EQ. 0) GO TO 999
   70 WRITE(6,7007)
      GO TO 999
   80 WRITE(6,7008)
  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-//- BUTTERWORTH--/
    A   - BWBNDP-,5X,-BAND PASS-/
    B   - BWLOWP-,5X,-LOW PASS-/
    C   - BWHIP-,6X,-HIGH PASS-/
    D   - BWBSTP-,5X,-BAND STOP-/
    E   /- TCHEBYSHEFF--/
    F   - CHBNDP-,5X,-BAND PASS-/
    G   - CHLOWP-,5X,-LOW PASS-/
    H   - CHHIP-,6X,-HIGH PASS-/
    I   - CHBSTP-,5X,-BAND STOP-/
    J   /- SYNCHRONOUSLY TUNED--/
    K   - SYNBP-,6X,-BAND PASS-/
    L   - SYNLP-,6X,-LOW PASS-/
    M   - SYNHP-,6X,-HIGH PASS-)
7004 FORMAT(/,10X,-DEMODULATORS--//
    A   - FMDEMO-,5X,-FM DEMODULATOR-/
    B   - AMDEMO-,5X,-AM DEMODULATOR-/
    C   - PHDEMO-,5X,-PHASE DEMODULATOR-)
7005 FORMAT(/,10X,- MISCELLANEOUS -//,
    A   - FRQMUL-,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,BLOCKNAME   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-,10X,-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, ELFIND, ADJN, PRTFAC

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 carred 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,DELF,DELT,PD
      COMMON /CWORD/ WORD(10)
      COMMON /CFLGS/ PDFLG,ARFLG
      LOGICAL PDFLG,ARFLG
      NFLG = 0
      NSET = N
      CALL PERIOD
      IF(ARFLG) GO TO 50
      NSET = N
      WRITE(6,7000) PD,DELF
   10 WRITE(6,7001) N,IGAM,DELT
   20 CALL FETCH(WORD,L,NBAD)
      CALL ELFIND(WORD,LTYP)
      IF(LTYP .EQ. 30) GO TO 900
      IF(LTYP .EQ. 31) GO TO 30
      IF(LTYP .EQ. 32) GO TO 40
      IF(LTYP .EQ. 8) STOP
      WRITE(6,7002)
      GO TO 20
```

```
   30 WRITE(6,7003)
      GO TO 20
   40 N = WORD(2)
      NFLG = 0
      CALL ADJN
      IF(N .GE. NSET) GO TO 10
      NFLG = 1
      GO TO 900
   50 IF(NSET .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,NSET
      WRITE(6,7003)
      GO TO 20
  990 N = NSET
  999 PDFLG = .FALSE.
      ARFLG = .TRUE.
      RETURN
 7000 FORMAT(/,- PERIOD = -,1PE10.3,- SECONDS,-,-DELTA-F = -,
     A         E10.3)
 7001 FORMAT(- N = -,I6,-, IGAM = -,I2,- , DELTA-T = -
     1         ,1PE10.3/- IS THIS SATISFACTORY-)
 7002 FORMAT(- INPUT MEANINGLESS * ENTER YES, NO, OR N,VALUE-)
 7003 FORMAT(- ENTER N, VALUE-)
 7004 FORMAT(- N = -,I6,- UNACCEPTABLE ** N MUST-
     1                 - BE-,I6,- TO MEET NYQUIST CRITERION-)
 7005 FORMAT(- SOURCE FREQUENCIES REQUIRE THE ARRAY SIZE -
     1         -TO BE -,I9//- 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:

```
      SUBROUTINE PERIOD
C * * PERIOD EXAMINES NFR FREQUENCIES (MAX 6) IN ARRAY FR,
C * * FACTORS EACH INTO PRIME FACTORS,CONSTRUCTS DELF AS THE
C * * GREATEST COMMON FACTOR$ COMPUTES ARRAY SIZE N AND IGAM
C * * TO SATISFY THE NYQUIST CRITERION AND FFT REQUIREMENTS
C * * COMPUTES THE TOTAL PERIOD PD AND SAMPLING PERIOD DELT.
      COMMON N,IGAM,DELF,DELT,PD
      COMMON/CFREQ/ NFR,FR(6)
      DIMENSION IA(30,6), IOUT(30), ICT(6)
      LOGICAL FLAG
      FLAG = .TRUE.
      GO TO 5
      ENTRY PRTFAC
      FLAG = .FALSE.
```

```
    5 MAXFAC = 0
      DO 100 I = 1,NFR
C * * CONVERT ITH FREQ TO INTEGER.
      IFR = FR(I)
C * * CLEAR ITH COL OF FACTOR ARRAY IA.
      DO 10 J = 1,30
   10 IA(J,I) = 0
      J = 1
      ITEST = 2
      IDEL = 1
C * * FACTOR ITH FREQ INTO PRIME FACTORS.
      GO TO 40
   30 ITEST = ITEST + IDEL
      IDEL = 2
   40 IF(IFR .EQ. 1) GO TO 90
      ALIM = SQRT(IFR)
      IF(ITEST .GT. ALIM) GO TO 60
   50 IF(MOD(IFR,ITEST) .NE. 0) GO TO 30
      IA(J,I) = ITEST
      J = J + 1
      IF(J .GE. 30) GO TO 990
      IFR = IFR / ITEST
      GO TO 50
   60 IA(J,I) = IFR
   90 IF (J .GT. MAXFAC) MAXFAC = J
  100 CONTINUE
      IF (FLAG) GO TO 102
      LIM = MAXFAC + 1
      DO 101 I = 1,LIM
  101 WRITE(6,711) I,(IA(I,J),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(1,I) .GT. IMAX) IMAX = IA(1,I)
C * * ADVANCE ALL ARRAY POINTERS TO A FACTOR .GE. IMAX.
  135 DO 200 I = 1,NFR
  140 J = ICT(I)
      IF(IA(J,I) .EQ. 0) GO TO 400
      IF(IA(J,I) .GE. IMAX) GO TO 200
      ICT(I) = J + 1
      GO TO 140
  200 CONTINUE
```

```
C * * IF ALL POINTED FACTORS ARE NOT IMAX, ADVANCE IMAX.
  210 DO 220 I = 1,NFR
      J = ICT(I)
      IF(IA(J,I) .NE. IMAX) GO TO 300
  220 CONTINUE
C * * ALL FACTORS ARE IMAX. TRANSFER TO IOUT, ADVANCE ALL
C * * POINTERS ONE STEP, AND RETEST FOR COMMON FACTOR.
      IOUT(IOCT) = IMAX
      IOCT = IOCT + 1
      DO 230 I = 1,NFR
  230 ICT(I) = ICT(I) + 1
      GO TO 210
  300 IMAX = IA(J,I)
      GO TO 135
  400 IOCT = IOCT - 1
      DELF = 1.
C * * IF NO FACTORS IN IOUT, DELF = 1. OTHERWISE DELF =
C * * PRODUCT OF ALL PRIMES IN IOUT.
      IF(IOCT .EQ. 0) GO TO 415
      DO 410 I = 1,IOCT
  410 DELF = DELF*IOUT(I)
  415 PD = 1. /DELF
      FMAX = 0.
      DO 420 I = 1,NFR
  420 IF(FR(I) .GT. FMAX) FMAX = FR(I)
C * * SAMPLING PERIOD TO MEET NYQUIST CRITERION.
      N = 2. * PD * FMAX + .5
C * * ARRAY SIZE AND SAMP PERIOD (DELT) TO SATISFY FFT.
      CALL ADJN
      GO TO 999
  990 WRITE(6,700) I
      STOP
  700 FORMAT(//1H ,-FREQUENCY -,I2,-HAS MORE THAN 29 FACTORS-/)
  711 FORMAT(1H ,6(1X,I6))
  999 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 beyong the $\pm\pi$ range thus reconstructing the phase deviation due to the angle modulation.

Program Listing:

```
      SUBROUTINE PHDEMO(A)
      COMMON N,IGAM,DELF,DELT,PD,CARRFQ
      COMMON/CDATA/ JCTR,DATA(200)
      COMPLEX A(1)
      PI2 = 6.2831853
      J = 0.
      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
      IF0 = F0 / DELF + .5
      NSTART = N2 + 1
      NSTOP1 = N - IF0 + 1
      DO 11 I = NSTART,N
   11 A(I - IF0) = A(I)
      DO 12 I = NSTOP1,N
   12 A(I) = (0., 0.)
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, 0.)
      A(1) = CMPLX(AIMAG(A(1)), 0.)
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, CHHIP, CHBSTP, SYNBP, SYNLP, SYNHP, 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.

Program Listing:

```
      SUBROUTINE PROCES(I,A)
      COMPLEX A(1)
      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 SIGGEN(A)
      RETURN
    2 CALL AMP(A)
      RETURN
    3 CALL BWBNDP(A)
      RETURN
    4 CALL BWLOWP(A)
      RETURN
    5 CALL BWHIP(A)
      RETURN
    6 CALL BWBSTP(A)
      RETURN
    7 CALL CHBNDP(A)
      RETURN
    8 CALL CHLOWP(A)
      RETURN
    9 CALL CHHIP(A)
      RETURN
   10 CALL CHBSTP(A)
      RETURN
   11 CALL SYNBP(A)
      RETURN
   12 CALL SYNLP(A)
      RETURN
   13 CALL SYNHP(A)
      RETURN
   14 CALL FRQMUL(A)
   15 RETURN
   16 CALL IDLMUL(A)
      RETURN
   17 CALL LIM(A)
   18 RETURN
   19 CALL FMDEMO(A)
      RETURN
   20 CALL PHDEMO(A)
   21 RETURN
   22 CALL AMDEMO(A)
      RETURN
   23 CALL FLATSP(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(DBMAX,MAXSCL)
C * * *  THIS SUBROUTINE ESTABLISHES ORDINATE SCALING FOR
C        THE REMOTE SPECTRUM PLOTTER.
      IF(DBMAX.LE.0.) GO TO 10
      MAXSCL = 0
    1 MAXSCL = MAXSCL + 10
      DIFF = DBMAX - MAXSCL
      IF(DIFF.GT.0.) GO TO 1
      GO TO 999
   10 MAXSCL = 0
   11 MAXSCL = MAXSCL - 10
      DIFF = (DBMAX - 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:

```
      SUBROUTINE SIGGEN(A)
      COMMON/CDATA/ JCTR,DATA(200)
      COMMON/CDOM/ DOMFLG
      COMMON N,IGAM,DELF,DELT
      COMPLEX A(1)
      LOGICAL DOMFLG
      DOMFLG = .TRUE.
      FC = DATA(JCTR)
      FMOD = DATA(JCTR + 1)
      PCTAM = DATA(JCTR + 2)
      PKPHDV = DATA(JCTR + 3)
      PKFRDV = DATA(JCTR + 4)
      AMPS = DATA(JCTR + 5)
      PI2 = 6.2831853
      WC = PI2 * FC
      WM = PI2 * FMOD
      BETAAM = PCTAM / 100.
      IF(PKPHDV .GT. .01 .AND. PKFRDV .GT. .01) GO TO 900
      IF(PKFRDV .GT. .01) GO TO 500
      DO 1 I = 1,N
      II = I - 1
      T = II * DELT
      COEF = AMPS * ( 1. + BETAAM * COS(WM * T))
      ANG = WC * T + PKPHDV * COS(WM * T)
    1 A(I) = CMPLX((COEF * SIN(ANG)), 0.)
      GO TO 999
  500 BETAFM = PKFRDV / FMOD
      DO 2 I = 1,N
      II = I - 1
      T = II * DELT
      COEF = AMPS * ( 1. + BETAAM * COS(WM * T))
      ANG = WC * T + BETAFM * SIN(WM * T)
    2 A(I) = CMPLX((COEF * SIN(ANG)), 0.)
      GO TO 999
  900 WRITE(6,101)
  101 FORMAT(1H ,-TIME FCN ERROR -- BOTH FR & PH MOD INDICES-,
     A                    - SPECIFIED-)
  999 RETURN
      END
```

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:

```
      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(10)
      LOGICAL PDFLG
      JCTR = ITYP(NBLK + 1,2)
      DO 10 I = 1,K1
   10 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. 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
```

29. Subroutine SYNBP

Called by: PROCES

Calls: FRQFCN

Commons: blank, CDATA

Entries: SYNLP, SYNHP

Description: SYNBP provides a model of a synchromously tuned bandpass filter. Entries SYNLP 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

$$n = \text{number of sections,}$$

$$f_p = \begin{cases} f/f'_{co} \text{ for low pass,} \\ - f'_{co}/f \text{ for high pass,} \\ (f - f_o)/f'_{co} \text{ for bandpass,} \end{cases}$$

$$f'_{co} = f_{co}/\Phi(n),$$

$$\Phi(n) = \sqrt{2^{1/n} - 1},$$

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:

```
      SUBROUTINE SYNBP(A)
      COMPLEX A(1),D,D1
      COMMON N,IGAM,DELF,DELT
      COMMON /CDATA/ 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 * DELF
      FP = SIGN(1.,F) * (ABS(F) - F0) / FCPR
      IF (FFLG) GO TO 30
      IF (ABS(FP) .LT. 1.E-16) GO TO 40
      FP = -1./FP
   30 IF (ABS(FP) .GT. TEST) GO TO 40
      D1 = CMPLX(1.,FP)
      D = D1**NR
      A(I) = A(I) / D
      GO TO 50
   40 A(I) = CMPLX(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:

```
      SUBROUTINE TELPLT(A,NST,NSP,NJUMP)
      COMPLEX A(1)
      DIMENSION IA(50)
      COMMON N,IGAM,DELF,DELT,PD
      CALL TIMFCN(A)
      BMAX = REAL(A(NST))
      BMIN = BMAX
      DO 1 I = NST,NSP,NJUMP
      B = REAL(A(I))
      IF(B.LT.BMIN) BMIN = B
    1 IF(B.GT.BMAX) BMAX = B
      IF((BMAX-BMIN).LT.1.E-30) GO TO 999
      DO 33 I = 1,50
   33 IA(I) = 1H
  100 WRITE(6,4)
    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(8X,2H 0,3X,2H.1,3X,2H.2,3X,2H.3,3X,2H.4,3X,2H.5,
     1   3X,2H.6,3X,2H.7,3X,2H.8,3X,2H.9,2X,3H1.0)
      WRITE(6,7)
    7 FORMAT(1H,8X,1HI,10(5H----I))
      DO 10 I = NST,NSP,NJUMP
      B = REAL(A(I))
      B = (B-BMIN)/(BMAX-BMIN)
      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(2X,I5,3H  I,50A1)
      GO TO 36
   34 WRITE(6,37) I
   37 FORMAT(2X,I5,3H  *)
   36 CONTINUE
   10 CONTINUE
      WRITE(6,11)
   11 FORMAT(6X,1HN)
      GO TO 900
  999 WRITE (6,9)
    9 FORMAT(1X,12HERROR FINISH)
  900 RETURN
      END
```

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,IGAM
      COMMON/CDOM/ DOMFLG
      LOGICAL DOMFLG
      IF(DOMFLG) GO TO 999
      CALL LFOLD(A,N)
      CALL FFT(A,IGAM, 1)
      DOMFLG = .TRUE.
      GO TO 999
      ENTRY FRQFCN(A)
      IF(.NOT. DOMFLG) GO TO 999
      CALL FFT(A,IGAM,-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 TELYTYPE PLOI OF IHE 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
```

```
C * * * *
      DBMAX = -1.E30
      IEND = NSP - NST + 1
      DO 1 I = 1,IEND
      DECTMP = 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 * * * *
C * * * *
      WRITE(6,2) IEND
    2 FORMAT(/5X,8HNSIZE = ,I5/)
C * * * *
      CALL SCALE(DBMAX,MAXSCL)
C * * * *
C    THE ORDINATE WILL VARY FROM(MAXSCL-50) DB UP
C    TO MAXSCL DB.
      DO 33 I = 1,50
   33 IA(I) = 1H
      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)
  200 FORMAT(2X,14HFREQUENCY (HZ),9X,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,I4,4(6X,I4),5X,I4)
      WRITE(6,8)
    8 FORMAT(9X,1HI,5(10H----+----I))
```

```
      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 + I - 1))
      IF(DECTMP.LT.1.E-30) DECTMP = 1.E-30
      B = 20. * ALOG10(DECTMP)
      M = 50 + B - MAXSCL
      J = I - 1
      XJ = J
      FREQ = FLOPRT + XJ * DELF1
      IF(M.LT.0) GO TO 50
      IF(M.EQ.0) GO TO 34
      DO 1515 II = 1,M
 1515 IA(II) = 1H-
      WRITE(6,35) FREQ,IA
      DO 1616 II = 1,M
 1616 IA(II) = 1H
   35 FORMAT(F8.3,2H I,50A1)
      GO TO 36
   34 WRITE(6,37) FREQ
   37 FORMAT(1X,F7.3,2H -)
      GO TO 36
   50 WRITE(6,51) FREQ
   51 FORMAT(1X,F7.3,2H I)
 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(1)
      COMMON N,IGAM,DELF
      CALL TIMFCN(A)
      WRITE (6,704)
      DO 10 I = N1,N2
   10 WRITE(6,700) I,A(I)
      GO TO 999
      ENTRY WRFF(A,FLO,FHI)
      CALL FRQFCN(A)
      NCTR = N/2 + 1
      FLOT = FLO - DELF/10.
      FHIT = FHI + DELF/10.
      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.E30
      T = CABS(A(I))
      IF (T .LT. 1.E-30) GO TO 15
      THETA = 57.29578*ATAN2(AIMAG(A(I)),REAL(A(I)))
   15 DB = 1.E30
      IF (T .GT. 0.) DB = 20.*ALOG10(T)
      WRITE(6,701) I,F,A(I),T,DB,THETA
   20 CONTINUE
  999 WRITE(6,703)
  700 FORMAT(1H ,I5,2(1PE10.3,2X))
  701 FORMAT(1H ,I5,1PE12.4,3(2X,0PF8.3),2X,F7.2,2X,F6.1)
  702 FORMAT(/-  LINE-,5X,-FREQ-,9X,-REAL-,6X,-IMAG-,6X,
     1  -MAG-,6X,-DB-,5X,-PHASE-/)
  703 FORMAT(/)
  704 FORMAT(/-  LINE-,6X,-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.

```
      COMPLEX A(1024)
      COMMON N,IGAM,DELF,DELT,PD,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/CFLGS/ PDFLG,ARFLG
      LOGICAL PDFLG, ARFLG
      JCTR = 1
      ITYP(1,2) = JCTR
      NFR = 0
      NBLK = 0
      IBLK = 0
      PDFLG = .FALSE.
      ARFLG = .FALSE.
      WRITE(6,7006)
    1 DO 2 I = 1,12
    2 WORD(I) = 4H
      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,
     1       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. 1H ) GO TO 12
      N1 = WORD(3)
      N2 = WORD(4)
      GO TO 15
   12 WRITE(6,7004)
      READ(5,7000) N1,N2
   15 CALL WRTF(A,N1,N2)
      GO TO 1
   20 IF (WORD(4) .EQ. 1H ) GO TO 22
      FRLO = WORD(3)
      FRHI = WORD(4)
      GO TO 25
   22 WRITE(6,7005)
      READ(5,7000) 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. 1H ) GO TO 32
      NST = WORD(3)
      NSP = WORD(4)
      NJUMP = WORD(5)
      IF (NJUMP .EQ. 1H ) NJUMP = 1
      GO TO 35
```

```
   32 WRITE(6,7007)
      READ(5,7000) 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,2(I3,3X))
   60 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
```

```
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 STRDTA(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 STRDTA(2,1,2)
      NTYP = 16
      GO TO 600
  240 GO TO 430
C * * FM DEMODULATOR
  250 CALL STRDTA(1,0,0)
      NTYP = 19
      GO TO 600
C * * PHASE DEMODULATOR
  260 CALL STRDTA(1,0,0)
      NTYP = 20
      GO TO 600
C * * AMPLIFIER
  270 CALL STRDTA(1,0,0)
      NTYP = 2
      GO TO 600
C * * LIMITER
  280 CALL STRDTA(3,0,0)
      NTYP = 17
      GO TO 600
  290 NOUT = WORD(3)
      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)
  292 CONTINUE
      IBLK = NOUT
  295 WRITE(6,7002) IBLK
      GO TO 1
  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 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()
C7000 FORMAT(3G.0)
 7001 FORMAT(- * * ERROR * * LARGEST BLOCK NO IS -,I2,- * *-)
 7002 FORMAT(- PROCESSING COMPLETE THRU BLOCK -,I2)
 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 BUFF1,BUFF2,BUFF3,BCDFPT,TITLE,WORD,E
      DIMENSION TITLE(12)
      COMMON/CFETCH/ BUFF2(6),BUFF1(80)
      DIMENSION WORD(1)
      EQUIVALENCE (BUFF3,BUFF4)
      DATA APOST,BLANK,COMMA,DECPT,EQUAL,MINUS,NINE,NZ,PLUS
     1    / 1H- ,1H  ,1H,,1H. ,1H= ,1H- ,1H9 ,1H0 ,1H+ /
      DATA LPAREN,RPAREN
     1    /  1H( ,  1H) /
      DATA E / 1HE /
      L=0
      NCOLS = 80
      NBAD = 1
C
C         FETCH IS A FREE-FIELD INPUT SUBPROGRAM WHICH RETURNS
C         THE INPUT DATA IN LL CONSECUTIVE CELLS OF THE
C         ARRAY WORD. HOLLERITH IS TRUNCATED TO 6 CHARS.
    1 CONTINUE
      READ  (5,1001,END = 107)  ( BUFF1(I),I=1,80 )
      WRITE (6,1005) ( BUFF1(I),I=1,80 )
   20 M = 0
    2 K = 0
      N = 0
      NCOMMA = 0
      DO 3 I = 1,6
    3 BUFF2(I) = BLANK
C
    4 IF ( M-NCOLS ) 40,100,100
C   *      EXAMINE EACH COLUMN, REMOVE BLANKS, AND TEST FOR
C          SEPARATORS.
   40 M = M+1
      TEST = BUFF1(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 ) 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.
```

```
    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 ) 55,30,55
   55 IF ( TEST - DECPT ) 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 = BCDFPT( BUFF2,N )
      IF ( N ) 106,106,8
    8 WORD(L+1) = BUFF3
      GO TO 91
    9 ENCODE(4,1002,WORD(L+1)) (BUFF2(I),I=1,4)
      ENCODE(2,1002,WORD(L+2)) (BUFF2(I),I=5,6)
      L = L + 1
   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.
      IF ( M-NCOLS ) 2,10,10
   10 IF ( TEST - COMMA ) 11,1,11
   11 LL = L
      RETURN
C   *      MOVE A TITLE CARD INTO THE TITLE ARRAY.
   30 ENCODE(48,1001,TITLE) (BUFF1(I),I = 1,48)
      GO TO 1
C   *      A CARD IMAGE HAS BEEN PROCESSED.  IF THE LAST
C          NON-BLANK SYMBOL WAS A COMMA (NCOMMA=0), READ
C          THE NEXT CARD. OTHERWISE THERE IS INFORMATION
C          IN *BUFF2* TO BE CONVERTED, AFTER WHICH, FETCH
C          WILL RETURN TO THE CALLING PROGRAM.
  100 IF ( NCOMMA ) 6,1,6
C   *      FETCH FOUND CONCOMITANT SEPARATORS OR A NUMBER WITH
C          MORE THAN 15 DIGITS AND COULD NOT CONTINUE.
  106 WRITE (6,2000)  (BUFF2(I),I=1,6)
      NBAD = 0
      RETURN
  107 STOP
 1001 FORMAT(80A1)
 1002 FORMAT(6A1)
 1005 FORMAT(1X80A1)
 2000 FORMAT(25H0**  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,
     B          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(6,7101)
      IF (FLG) GO TO 999
   10 WRITE(6,7010)
      WRITE(6,7102)
      IF (FLG) GO TO 999
   11 WRITE(6,7011)
      WRITE(6,7102)
      IF (FLG) GO TO 999
   12 WRITE(6,7012)
      WRITE(6,7101)
      IF (FLG) GO TO 999
   13 WRITE(6,7013)
      WRITE(6,7101)
      WRITE(6,7103)
      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
```

```
   17 WRITE(6,7017)
      WRITE(6,7101)
      IF (FLG) GO TO 999
   18 WRITE(6,7018)
      WRITE(6,7102)
      IF (FLG) GO TO 999
   19 WRITE(6,7019)
      WRITE(6,7102)
      IF (FLG) GO TO 999
   25 WRITE(6,7025)
      WRITE(6,7107)
      IF (FLG) GO TO 999
   34 WRITE(6,7034)
      WRITE(6,7107)
      IF (FLG) GO TO 999
   26 WRITE(6,7026)
      WRITE(6,7107)
      IF (FLG) GO TO 999
   21 WRITE(6,7021)
      WRITE(6,7105)
      IF (FLG) GO TO 999
   23 WRITE(6,7023)
      WRITE(6,7106)
      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
 7000 FORMAT()
 7001 FORMAT(1X,A4,A2,- IS NOT AN INPUT COMMAND-)
 7009 FORMAT(/- BWBNDP, F0, BW, NR-)
 7010 FORMAT(/- BWLOWP, FC, NR-)
 7011 FORMAT(/- BWHIP, FC, NR-)
 7012 FORMAT(/- BWBSTP, F0, BW, NR-)
 7013 FORMAT(/- CHBNDP, F0, BW, NR, EPSDB-)
 7014 FORMAT(/- CHLOWP, FC, NR, EPSDB-)
 7015 FORMAT(/- CHHIP, FC, NR, EPSDB-)
```

```
7016 FORMAT(/- CHBSTP, FO, BW, NR, EPSDB-)
7017 FORMAT(/- SYNBP, FO, BW, NR-)
7018 FORMAT(/- SYNLP, FC, NR-)
7019 FORMAT(/- SYNHP, FC, NR-)
7020 FORMAT(/- SIGGEN, FO, FMOD, AM, PM, FM, A-)
7021 FORMAT(/- FRQMUL-)
7023 FORMAT(/- IDLMUL, ALO, FLO-)
7025 FORMAT(/- FMDEMO, FO-)
7026 FORMAT(/- PHDEMO, FO-)
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(- FO = 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(- FO = CARRIER FREQ, HZ-/
    A        - FMOD = MODULATION FREQ, HZ-/
    B        - 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(- FO = 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,4HCPLO,2HTF,4HPRIM,2HEF,4HENDO,2HFJ,
     C 4HBWBN,2HDP,4HBWLO,2HWP,4HBWHI,2HP  ,4HBWBS,2HTP,
     D 4HCHBN,2HDP,4HCHLO,2HWP,4HCHHI,2HP  ,4HCHBS,2HTP,
     E 4HSYNB,2HP  ,4HSYNL,2HP  ,4HSYNH,2HP  ,4HSIGG,2HEN,
     F 4HFRQM,2HUL,4H    ,2H  ,4HIDLM,2HUL,4H    ,2H  ,
     G 4HFMDE,2HMO,4HPHDE,2HMO,4HAMP ,2H  ,4HLIM ,2H  ,
     H 4HBLOC,2HK ,4HYES ,2H  ,4HNO  ,2H  ,4HN   ,2H  ,
     I 4H    ,2H  ,4HAMDE,2HMO,4HFLAT,2HSP,4HLIST,2HCO,
     J 4HCIRC,2HUI,4HINPU,2HTF,4HDELE,2HTE,4HINSE,2HRT,
     K 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)
      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(6,7020) WORD(3)
      GO TO 999
   20 WRITE(6,7002)
      IF (I .EQ. 0) GO TO 999
   30 WRITE(6,7003)
      IF (I .EQ. 0) GO TO 999
   40 WRITE(6,7004)
      IF (I .EQ. 0) GO TO 999
   50 WRITE(6,7005)
      IF (I .EQ. 0) GO TO 999
   60 WRITE(6,7006)
      IF (I .EQ. 0) GO TO 999
   70 WRITE(6,7007)
      GO TO 999
   80 WRITE(6,7008)
  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-//- BUTTERWORTH--/
     A   - BWBNDP-,5X,-BAND PASS-/
     B   - BWLOWP-,5X,-LOW PASS-/
     C   - BWHIP-,6X,-HIGH PASS-/
     D   - BWBSTP-,5X,-BAND STOP-/
     E   /- TCHEBYSHEFF--/
     F   - CHBNDP-,5X,-BAND PASS-/
     G   - CHLOWP-,5X,-LOW PASS-/
     H   - CHHIP-,6X,-HIGH PASS-/
     I   - CHBSTP-,5X,-BAND STOP-/
     J   /- SYNCHRONOUSLY TUNED--/
     K   - SYNBP-,6X,-BAND PASS-/
     L   - SYNLP-,6X,-LOW PASS-/
     M   - SYNHP-,6X,-HIGH PASS-)
```

```
7004 FORMAT(/,10X,-DEMODULATORS--//
    A   - FMDEMO-,5X,-FM DEMODULATOR-/
    B   - AMDEMO-,5X,-AM DEMODULATOR-/
    C   - PHDEMO-,5X,-PHASE DEMODULATOR-)
7005 FORMAT(/,10X,- MISCELLANEOUS -//,
    A   - FRQMUL-,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,BLOCKNAME   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,LII ,ETC)-/
    I - CONTROL-,8X,-LIST CONTROL COMMANDS-/
    J - NSIZE-,10X,-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,DELF,DELT,PD
      COMMON /CWORD/ WORD(12)
      COMMON /CFLGS/ PDFLG,ARFLG
      LOGICAL PDFLG,ARFLG
      NFLG = 0
      NSET = N
      CALL PERIOD
      IF(ARFLG) GO TO 50
      NSET = N
      WRITE(6,7000) PD,DELF
   10 WRITE(6,7001) N,IGAM,DELT
   20 CALL FETCH(WORD,L,NBAD)
      CALL ELFIND(WORD,LTYP)
      IF(LTYP .EQ. 30) GO TO 900
      IF(LTYP .EQ. 31) GO TO 30
      IF(LTYP .EQ. 32) GO TO 40
      IF(LTYP .EQ. 8) STOP
      WRITE(6,7002)
      GO TO 20
   30 WRITE(6,7003)
      GO TO 20
   40 N = WORD(3)
      NFLG = 0
      CALL ADJN
      IF(N .GE. NSET) GO TO 10
      NFLG = 1
      GO TO 900
   50 IF(NSET .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,NSET
      WRITE(6,7003)
      GO TO 20
  990 N = NSET
  999 PDFLG = .FALSE.
      ARFLG = .TRUE.
      RETURN
 7000 FORMAT(/,- PERIOD = -,1PE10.3,- SECONDS,-,-DELTA-F = -,
     A          E10.3)
 7001 FORMAT(- N = -,I6,-, IGAM = -,I2,- , DELTA-T = -
     1          ,1PE10.3/- IS THIS SATISFACTORY-)
 7002 FORMAT(- INPUT MEANINGLESS * ENTER YES, NO, OR N,VALUE-)
 7003 FORMAT(- ENTER N, VALUE-)
 7004 FORMAT(- N = -,I6,- UNACCEPTABLE ** N MUST-
     1                 - BE-,I6,- TO MEET NYQUIST CRITERION-)
 7005 FORMAT(- SOURCE FREQUENCIES REQUIRE THE ARRAY SIZE -
     1          -TO BE -,I9//- PRIME FACTORS ARE-/)
 7006 FORMAT(//- RUN IS BEING TERMINATED-)
      END
```

```
      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(12)
      LOGICAL PDFLG
      JCTR = ITYP(NBLK + 1,2)
      DO 10 I = 1,K1
   10 DATA(JCTR + I -1) = WORD(I + 2)
      JCTR = JCTR + K1
      IF(K2 .EQ. 0) GO TO 999
      PDFLG = .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
```