A FLEXIBLE USER ORIENTED APPROACH TO COMMUNICATIONS SYSTEMS SIMULATION

M. Fashano M.C. Austin
Hughes Aircraft Company
Space and Communications Group

ABSTRACT

An advanced software/hardware computer system developed for the simulation of communications systems is described. This user oriented system allows for flexible and efficient modeling and simulation of complex communications systems. Excellent agreement between simulation and measured results has consistently valididated the simulation approach.

INTRODUCTION

The role of simulation in the design and analysis of communications systems has been expanding as these systems become more complex in both their design and required performance. One of the prime objectives of simulation is to produce a better system with fewer design cycles while providing insight and confidence in its performance. Simulation provides an evaluation tool to aid in selecting between various hardware implementation concepts. Parametric studies also provide a basis for selecting realistic parameter specifications. Simulations may also generate data (e.g., waveforms, spectrums) to aid in the testing and verification of a system and to evaluate sensitivity to various system parameters.

In the past, an advanced simulation software package known as SYSTID (References 1-6) has been routinely used at HUGHES for communications systems simulation on large scale, general purpose computers. SYSTID provides the desirable characteristics of a simulation system: a user oriented interface for both system and model development; an extensive library of component models and performance estimators; uses minimal computer time; and products results that agree with hardware results. The increasing complexity of communications systems, however, has necessarily resulted in larger computational costs when large scale general computers are utilized.

With the recent availability of very capable minicomputers and array processors, a cost effective computational system becomes both economically and functionally feasible. This type of computational hardware, when coupled with a software system such as SYSTID, becomes an effective tool for communications systems simulation. Such an approach has been taken at HUGHES with the installation of a minicomputer coupled with an array processor and the simulation software. This system, known as the Communications Systems Simulator (CSS), and its use is introduced in this paper.

SYSTEM DESCRIPTION

Since the simulation software package known as SYSTID has been the subject of several publications (References 1-6), only a brief summary of SYSTID will be addressed. SYSTID is basically a language translator coupled with an element model library which aids the communications analyst in the modeling and simulation of complex communications systems. Its initial development began in 1967 specifically for application to communications systems and employs the complex envelope or analytic signal approach to modulated carrier communications systems. Utilization of SYSTID has significantly decreased the analyst's time required to solve a problem. This is due to both the user orientation of the language and the computational facilities available. SYSTID accepts as input a topological block diagram description of the particular system along with necessary parameters and run controls.

A SYSTID simulation program is capable of evaluating the performance of a complex system consisting of any number of filters and nonlinearities in any order. It is unconstrained by the number and types of elements other than limits imposed by computer size. The model libraries allow the user to specify such things as data input sequences, modulator types, modulator imperfections, filter types, non-linearities, noise sources, demodulator types and demodulator imperfections. System calibration models such as noise bandwidth, group delay, power measurements, and other models that expand the direct applicability and flexibility of the software system as a whole are included in the model libraries.

SYSTID offers the user a great deal of flexibility in the representation of system elements. Any system element that can be represented by a library model or a FORTRAN routine can be used as an element in the system description. An element may be defined as:

1) a SYSTID library model
2) a user written SYSTID model
3) an arithmetic expression involving any constant, variable, intrinsic SYSTID parameter, FORTRAN library function, SYSTID library function, model node, or any user supplied FORTRAN function

In addition, FORTRAN declaration and executable statments may be intermixed with the topological problem description.

SYSTID has the additional flexibility of linking to user defined postprocessing routines. This allows the user to access the time histories of any node or variable for further processing or analysis. Separate utility routines are also available to perform input and output processing for the user. The performance analysis estimators (e.g., spectra, distortion, etc.) are usually implemented as postprocessors.

The flexibility of SYSTID is in part attained by designing the program to execute as a multipass processor in a batch or demand (timesharing) mode of operation. Figure 1 presents one form of describing the information flow pertaining to SYSTID. The user normally describes a problem from a terminal using an editor to create a disc file for input to the translator. The translator reads the user topology and parameter description and proceeds to generate the corresponding FORTRAN routines for the specific problem. In this phase the translator checks for input errors such as erroneous model references, Inconsistencies and typographical errors in which case appropriate error messages are issued. Following translation, the FORTRAN routines are compiled and loaded with the model and system libraries to form the executable module which is then started. Results of the simulation may be optionally saved on disc data files for access by interactive output modules or as inputs to other simulation or analysis processes.

The user, because of the two phase aspect of SYSTID, has available several techniques for controlling computer runs and ensuring that the most effective use is made of both his time and computer time. One method is to save the results of the translator (FORTRAN routine and/or executable modules) for subsequent reruns with alternate parameters. A rerun would simply entail a load-go operation with the alternate parameter values provided at run time.

The SYSTID model library contains a large set of routines, written either in FORTRAN or SYSTID, which have been stored on a library file and cataloged in the SYSTID directory. The user at any time can modify or replace the library and directory. Thus, every user can create his own library as it pertains to his problem. One useful characteristic of SYSTID is the capablility of nesting models, that is, any model (or system) can reference any other model(s) other than itself. This nesting feature provides the user with the tools necessary to build an extended library based upon a canonic set of models. An example might be a receiver that is used in several systems, where the receiver consists of a downconverter, prodetection filter, demodulator and postdetection filter. Another specialized nested model might be a phase-lock-loop.

Most of the models in the library have been developed iteratively with their hardware conterparts, making them as realistic as necessary for their simulation role. Also, most of the models have been implemeted for both complex envelope and direct simulations. Complex envelope modeling requires an analytic signal where the signals are represented as the envelope of an arbitrarily translated or removed carrier, whereas, for direct modeling the signals are not translated and the carrier is retained in the simulation.

It should be apparent that use of a comprehensive operating system is implied in Figure 1. Historically, SYSTID simulations have been performed on large general purpose (i.e., expensive) computers such as UNIVAC 1108 and IBM 370 type machines. However, Hughes has recently configured a minicomputer coupled with an array processor specifically for performing communications and signal processing system simulations.

Figure 2 shows the current hardware configuration of the Communications Systems Simulator (CSS). The CSS consists of a moderately sized PRIME 400 computer with 600 megabytes of disc storage, seven directly connected and two dial-up terminals, a graphics terminal, a data tablet, printer and plotter peripherals, and a moderately sized Floating Point Systems AP-120B array processor. The PRIME's timesharing, multi-user operating system (PRIMOS IV) is very comprehensive and includes editors, compilers, loaders, disc file system, RJE capability, networking, batch processing, etc. and is similar in function to the large scale systems. Also, it is a virtual system allowing up to 32 megabytes of direct address space per user, thus alleviating the software development and computer size constraints encountered in large simulations.

The speed of the PRIME 400, as with most other minicomputers, is less than the larger machines but still very respectable. The array processor, however, is capable of performing highly organized floating point operations at a rate of 12 million floating point operations per second (by comparison, the CDC 7600 does 5 megaflops as reported in IEEE Spectrum, August 1976) .... a very intimidating speed for such a little biege and green box. With this configuration, we are able to generate and manage our software, inputs and outputs on the PRIME, and direct the heavy computational load associated with simulations to the array processor (e.g., transforms, correlations, etc.).

As mentioned earlier, SYSTID simulations prior to the installation of the CSS were performed on large scale computers in the classical batch environement. Modification of SYSTID to utilize the array processor capability consists of modifying the model library and not the translator. However, to fully realize the minicomputer and array processor potential, the SYSTID translator and model libraries are In the process of being restructured and enhanced.

SIMULATION EXAMPLE

Figure 3 shows the signal flow block diagram of an 80 Mbps experimental linear test setup (discussed in Reference 7) capable of operating in either MSK or OKQPSK modes. For each configuration, MSK and OKQPSK, various channel filters were located between the transmitter and receiver. The filters were 7 pole Chebyshev designs differing only in bandwidth. The filter bandwidths were 40, 56, 80, and 113 Mhz. An additional 56 Mhz self-equalized filter was also constructed having an amplitude response very similar to the Chebyshev designs but with much less group delay variation across the filter passband. The above system was configured using various detection filters (i.e., integrate and dump and several passive 2 pole Butterworth filters with various bandwidths). The computer simulations were configured to match the hardware. To provide realism to the simulation results, the actual amplitude and phase responses of the channel filters were measure and used in the simulation. These measurements were read into the SYSTID filter model implemented in the frequency domain (FREEKY). Imperfections in the experimental modulator and demodulator (see Figure 3) were estimated and used in the simulation.

As an example of a simulation program, the SYSTID input needed to sumulate the OKQPSK configuration is given in Figure 4. Listed is the complete SYSTID definition of the above system’s topology and parameters, which illustrates the simplicity of the simulation setup and the inputs. The models used in the simulation are contained in the model library. The modulated carrier signal is output by the OKQPSK modulator which uses two independent pseudonoise (PN) bit sequences to represent the data modulation. Phase and amplitude imbalances, rise/fall times, skew offset between inphase and quadrature sequences, and bias distortion are controlled by model input parameters. One of several measured channel filters can be selected to represent the channel. Following the channel filter, noise is simulated simply by generating the signal at the bit detector output and evaluating the Gaussian distribution function with knowledge of the noise level. Measurements of the system time delay and the noise bandwidth of the receiver are automatically made and subsequently used in the bit error rate estimator. In the receiver, detection phase offset and timing offset bias are controlled by input parameters. The detection filter in this particular simulation description was an integrate and dump. However, any appropriate detection filter can be used by a simple statement change.

The accuracy of the simulation approach has been substantiated many times by the excellent agreement between hardware measurements and computer simulation (References 4, 8 and 9). As an example of this excellent agreement, Figure 5 shows the bit error rate (BER) comparison between the experimental test setup measurements and the simulation of the OKQPSK system illustrated in Figure 3. Other BER comparisons (using a setup similar to Figure 3) showing excellent agreement between measured and simulated results are given in Figures 6 through 8 respectively for the following:

TRANSMIT	RECEIVE
OKQPSK	no weighting, 2 pole detection filter
MSK	sinusoidal weighting, 2 pole detection filter
MSK	no weighting, 2 pole detection filter

Since the simulation allows for time waveform plots at any point in the system, it is possible to compare the breadboard and simulation detected waveforms. Figure 9, which corresponds to the MSK configuration with sinusoidal weighting in the receiver, shows a comparison between the breadboard and simulated detected waveforms at a receiver detection filter input. The photograph represents breadboard data modulated on an MSK carrier which was passed through an 80 Mhz channel filter and recovered in the receiver. Similarly, the plot depicts the processed simulation data which was filtered by the 80 Mhz channel filter model characterized by measured point-by-point data of the actual filter. The similarity between these two data waveforms ensures that the excellent agreement between measured and computed results was not caused by cancellation of various system imperfections in either the hardware or the simulation.

SUMMARY

A flexible, user oriented, cost effective software/hardware simulation system with a simple setup procedure, efficient computer run time and accurate system performance predictions has been developed and introduced. A comprehensive communications system model library exists, with most models developed/verified with hardware test results. Excellent agreement between experimental test data and computer simulation results has substantiated the accuracy of the simulation system and thus supports the usefullness of this system as a performance prediction tool.

ACKNOWLEDGEMENTS

A key contributor to the development of SYSTID was Dale Paynter. Jim Justiss and Dale Payntor played key roles in implementing the CSS. Also acknowledged is the support given by management of the Systems Laboratories, Space and Communications Group for providing up-to-date computional facilities.

REFERENCES

1. Final Report, SYSTID System Time Domain Simulation Program, NASA Johnson Spacecraft Center, Contract NAS9-10832, February 1971.

2. W.H. Tranter and C.T. Dawson, “A Flexible Tool for the Analysis of Communication Systems”, Proceedings of the 1972 National Telecommunications Conference, Paper 38c, December 1972.

3. Final Report, Advanced Communication System Time Domain Modeling Techniques, NASA Johnson Spacecraft Center, Contract NAS9-11743, August 1972.

4. M. Fashano, “Simulation of Satellite Communication Channels”, presented at the NATO Advanced Study Institute of New Directions in Signal Processing in Communications and Controls, August 1974.

5. Final Report, Design Analysis and Computer-Aided Performance Analysis of Shuttle Orbiter Electrical Power System, NASA/JSC, Contract NAS9-13779, October 1974.

6. M. Fashano, W.H. Tranter, and R.E. Zeimer, “Performance Evaluation Using SYSTID Time Domain Simulation”, International Conference on Communications, Paper 20a, June 1975.

7. D.F. Horwood and P.S. Angelo, “Experimental Evaluation of MSK and Offset Keyed QPSK Through Satellite Channels”, Proceedings of the 1978 International Telemetry Conference, November 1978.

8. M.C. Austin, “Wide-band Frequency-Shift Keyed Receiver Performance in the Presence of Intersymbol Interference”, IEEE Trans. Commun., Vol. COM-23, April 1975.

9. L.B. Milstein and M.C. Austin, “Performance of Noncoherent FSK and AM-FSK Systems with Postdetection Filtering”, IEEE Trans. Commun., Vol. COM-23, November 1975.

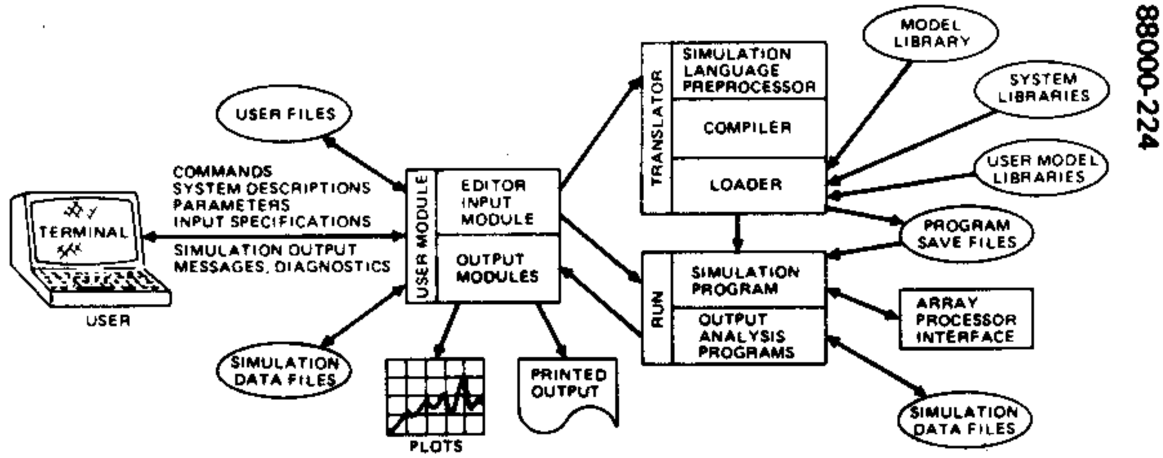


FIGURE 1. COMMUNICATIONS SYSTEMS SIMULATOR SOFTWARE CONFIGURATION

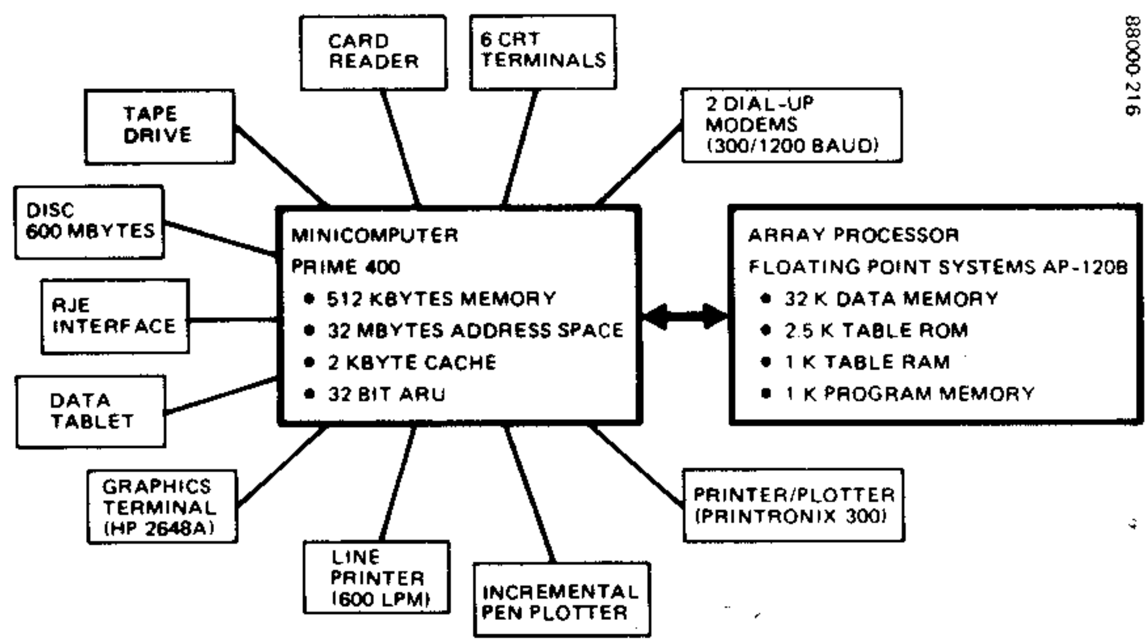


FIGURE 2 . COMMUNICATIONS SYSTEMS SIMULATOR HARDWARE CONFIGURATION

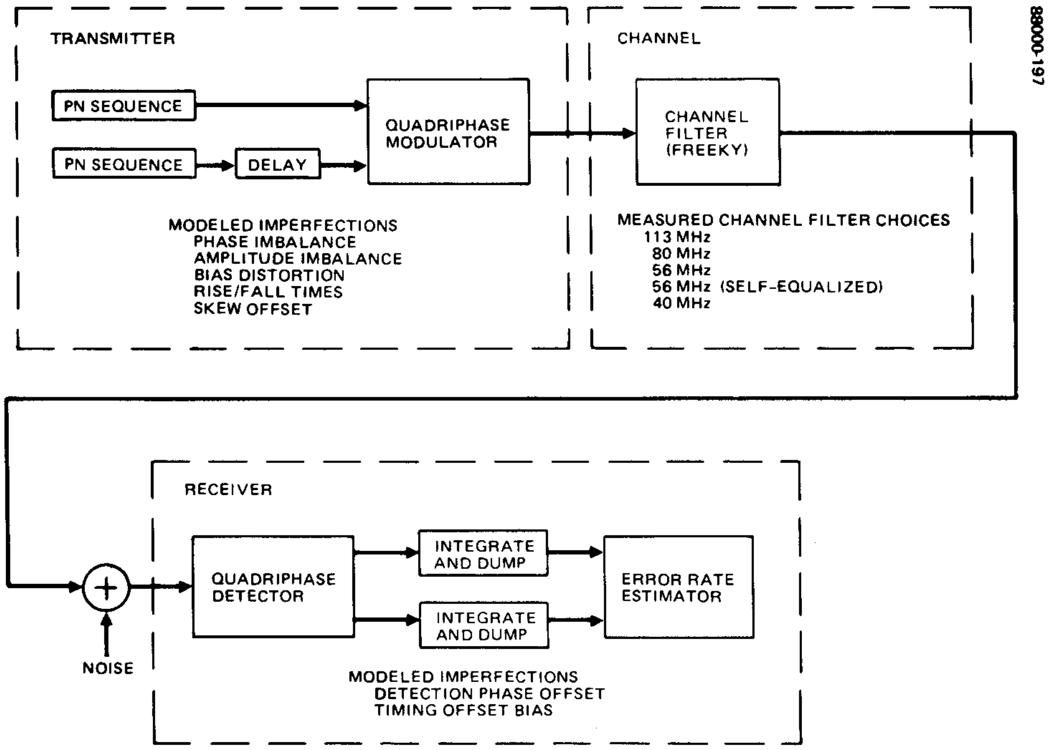


FIGURE 3. BLOCK DIAGRAM OF OKQPSK SYSTEM

```
SYSTEM:  OFFSET QPSK LINEAR CHANNEL

        DIMENSION END(3), PHI(3)
        DATA END/7.0, 2.0, 7.0/, PHI/0., 0., 1.0/

COMPLEX:  SCIN, SCOUT, DETIN
DEFAULT:  NB=64, NR=6, NSEQ=1, ENU=200., NSTB=20, ISLIP=3

SAVE (S-OKQPSK) DETIN, *DETIN                  - save for future use
SPECTRUM:  DETIN, *DETIN                       - generate spectrum of DETIN
PLOT:      DETIN                               - generate time waveform

       INITIALIZE THE RUN PARAMETERS

SET:   READ(1,*)BR, PIQ, THETA, RT, BIAS, SKEW
SET:   TB=1.0/BR                               - baud time
SET:   DT=TB/NSTB                              - simulation sample time
SET:   BI=BIAS*TB                              - bias distortion
SET:   RRT=RT*TB                               - rise & fall times
SET:   NDT=NSTB*NB                             - samples per pn sequence
SET:   AMPIQ=10.**(PIQ/20.)                    - amplitude imbalance

SET:   TSEQ=NB*TB                              - pn sequence duration
SET:   T1=TSEQ                                 - end time of noise bw calib.
SET:   T2=T1+10*TB                             - start of delay calib.
SET:   T3=T2+20*TB                             - end of delay calib.
SET:   T4=2*TSEQ                               - start ber estimation
SET:   SETTLE=T4                               - start time for plots
SET:   TSTOP=T4 + NSEQ*TSEQ + 10*TB            - simulation stop time

SET:   FT=1000.                                - carrier & traslation freq.
SET:   FL=FT-.5/DT                             - lower freq of filter window
SET:   FH=FT+.5/DT                             - upper freq of filter window

       GENERATE TWO PN DATA STREAMS

       10 P=PNPLS3(TB, NB, NR, 1, RRT, RRT, BI)
       20 Q=AMPIQ*PNPLS3(TB, NB, NR, ISLIP, RRT, RRT, BI)

       P, Q < QPSK MOD(FT, FT, THETA, SKEW) > SCIN
       SCIN < FREEKY(FL, FH, FT, NDT, PUP, PDOWN) > SCOUT
       SCOUT < ENB SIGNAL(TSEQ) > DETIN
       DETIN < QPSK ID DEMOD(TB, FT, FT, 0.) > POUT, QOUT

         PERFORM BIT ERROR RATE ESTIMATION

       30 EBD=PDOWN*TB/2.                      - enery per bit
       POUT, QOUT < BER ESTIMATOR(TB, NB, NSEQ, P, Q, ENU, END, T4, EBD,
                                 SKEW, T1, T2, T3, ENB, SDEL, PHI) > OUT
END
```

Figure 4 SYSTID input for OKQPSK Error Rate Estimation

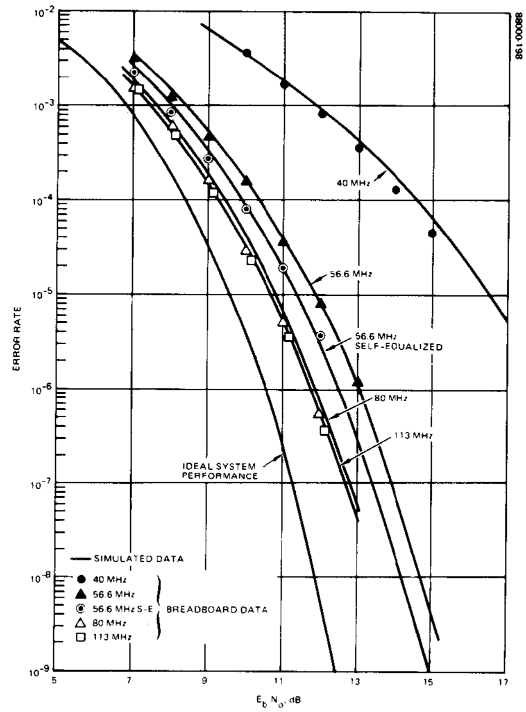
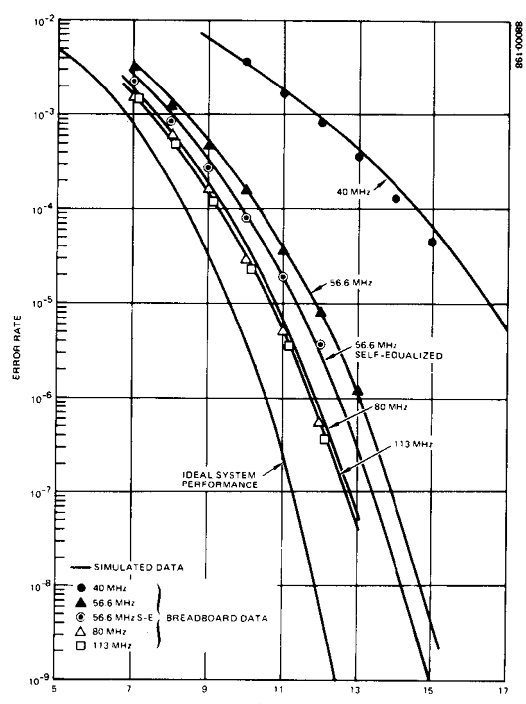


FIGURE 5. MEASURED AND SIMULATED BER PERFORMANCE FOR OKQPSK AND I&D DETECTION FILTER



FIGURED 6. MEASURED AND SIMULATED BER PERFORMANCE FOR OKQPSK AND TWO-POLE BUTTERWORTH DETECTION FILTER (BW = 21.5 MHZ)

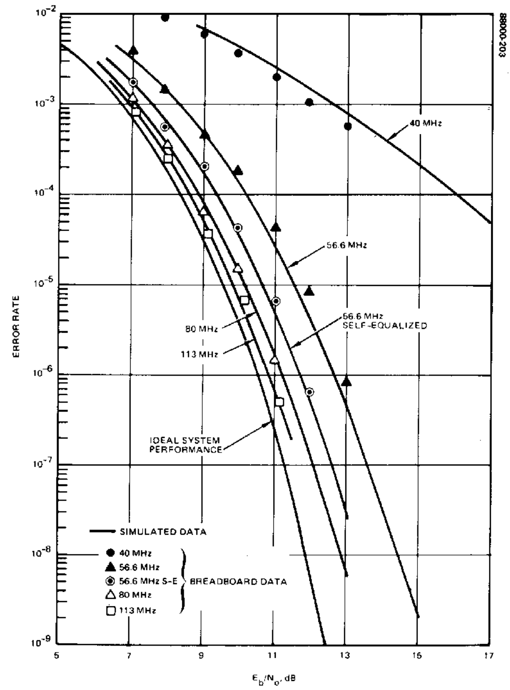


FIGURE 7. MEASURED AND SIMULATED BER PERFORMANCE FOR MSK (SINUSOIDAL WEIGHTING) AND TWO-POLE BUTTERWORTH DETECTION FILTER (BE = 21.5 MHZ)

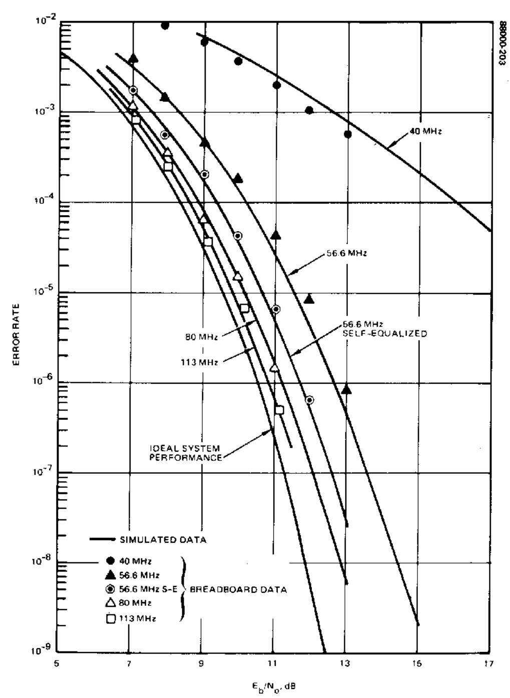


FIGURE 8. MEASURED AND SIMULATED BER PERFORMANCE FOR MSK (NO WEIGHTING) AND TWO-POLE BUTTERWORTH DETECTION FILTER (BW = 21.5 MHZ)

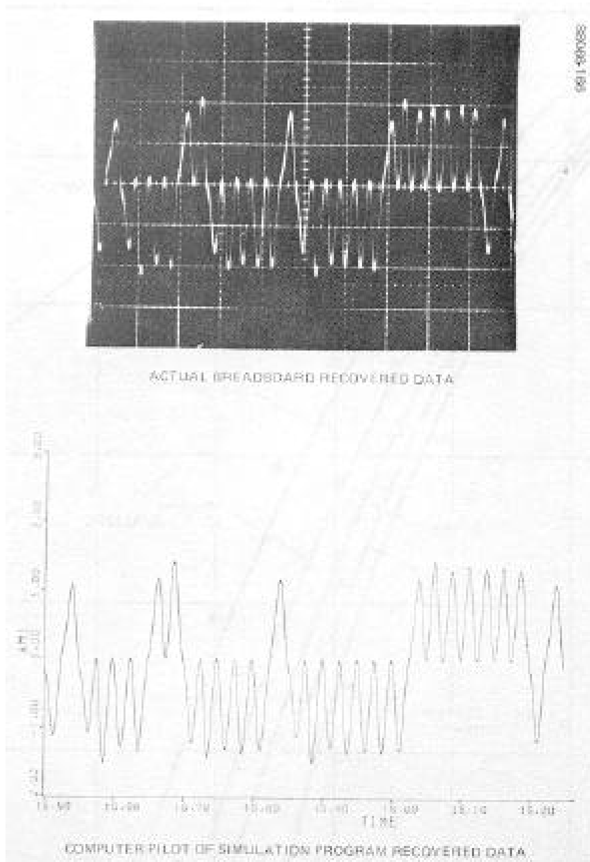


FIGURE 9. COMPARISON OF COMPUTED AND ACTUAL DATA WAVEFORMS FOR MSK (SINUSOIDAL WEIGHTING) WITH 80 MHZ CHANNEL FILTER