Technical Report 489

D. A. McNeill

A Simulation Facility for Communication Systems

27 December 1971

Prepared for the Department of the Navy
under Electronic Systems Division Contract F19628-70-C-0230 by

Lincoln Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Lexington, Massachusetts

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LINCOLN LABORATORY

A SIMULATION FACILITY FOR COMMUNICATION SYSTEMS

D. A. McNEILL

Group 66

TECHNICAL REPORT 489

27 DECEMBER 1971

LEXINGTON MASSACHUSETTS

The work reported in this document was performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. The work was sponsored by the Department of the Navy under Air Force Contract F19628-70-C-0230.

ABSTRACT

A simulation facility, consisting of a small general-purpose computer, associated peripherals, and extensive software, has been developed for analyzing communication systems. Presently, the facility is being used to develop an all digital Sanguine receiver and to gather performance statistics on the receiver. The facility and the simulated Sanguine transmitter and transmission channel necessary for operating the receiver are described.

Accepted for the Air Force
Joseph R. Waterman, Lt. Col., USAF
Chief, Lincoln Laboratory Project Office

CONTENTS

A SIMULATION FACILITY FOR COMMUNICATION SYSTEMS

I. INTRODUCTION

As part of its Navy Communications program, Lincoln Laboratory has undertaken the development and analysis of a receiver for an ELF communication system. The results of this effort are presented here and in a series of documents (Refs. 1 through 5). This report describes in detail the computer simulation facility which was used to develop the receiver.

An overriding feature of ELF communication systems is the high transmitter cost.[1] Hence, sophisticated receiver designs which improve performance and thereby minimize the required transmitter power become desirable. Such complex designs cannot always be analyzed mathematically, so that analysis must be supplemented by gathering performance statistics upon a simulated receiver. From Ref. 3 it is clear that sophisticated error-control coding and decoding can best be done digitally and, in fact, considering the data rates involved, can be done on a small computer. Therefore, a simulation facility designed around a small, general-purpose digital computer was developed for the receiver simulations, and a considerable amount of software support has been developed by Lincoln Laboratory specifically for the facility. However, since ELF noise processing results were needed before the simulation facility was available, most of the preliminary noise processing was done separately on an IBM 360 (Ref. 2).

In a Sanguine ELF communication system, a small computer can easily perform the necessary computations associated with decoding[3] and still have a substantial amount of computer time left over for other applications in the receiver. Although most receiver functions can be realized with analog components, there are advantages to an all digital realization. First and foremost is greater design flexibility. Other advantages are:

(a) Easier modification of parameter values,
(b) Greater precision and accuracy,
(c) Long-term stability,
(d) Enhanced integrability with other systems,
(e) Greater reliability,
(f) Ease of implementing self-checking features,
(g) Possible space, weight, and power savings.

Because of these advantages it is expected that a large portion of the operational receiver will be implemented on a computer.

The simulated receiver is all digital primarily to provide the flexibility necessary for examining different designs and secondarily to determine speed and accuracy requirements for an all digital receiver. It should be emphasized that the receiver is simulated on a computer not unlike one which could eventually be part of a receiver package. Such characteristics as short word length, limited memory, and roundoff errors had to be dealt with. Thus, the simulated receiver is implemented in a way that it could very well serve as a real operational receiver. With this approach the receiver can be examined more thoroughly than if the entire simulation were written, for example, in FORTRAN on a large-scale computer.

Although the simulation facility is designed as a test facility for an ELF receiver, its usefulness is not limited to this role. Other communication systems, either complete or in part, may also be analyzed. However, in order to be specific about some of the features of the facility,

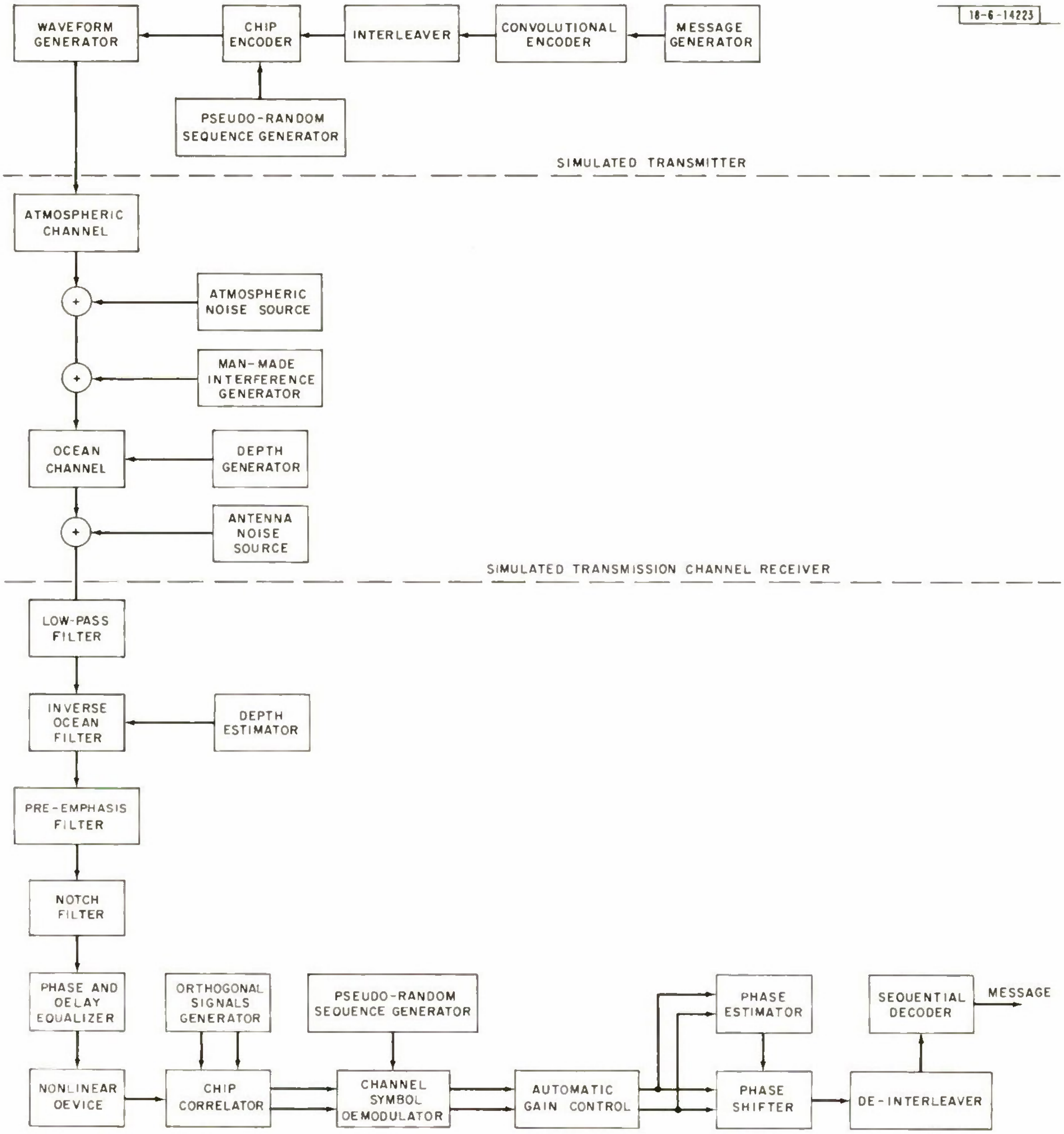

Fig. 1. Block diagram of simulated transmitter, transmission channel, and receiver.

reference will be made to the Sanguine ELF system. For that reason, Section II includes a summary and a block diagram of the Sanguine system; however, no attempt is made to describe it in detail.[1,3] The Control Program is the supervisor program of the facility and is described in Section III. Experimental procedures undertaken are described in Section IV. The support software is described in Section V. The simulated Sanguine transmitter and transmission channel necessary for operating the Sanguine receiver are described in Section VI. Appendix A lists the hardware components of the facility.

II. SANGUINE RECEIVER

The major functions of the Sanguine receiver are (a) filtering with nonlinear processing to reduce the effect of atmospheric noise, (b) phase tracking to allow phase coherent signaling, and (c) sequential decoding to provide the required reliability at a low signal-to-noise ratio. These functions are illustrated in a block diagram shown in Fig. 1. Until recorded signals at the receiver antenna terminals are recorded and made available, it is necessary to simulate the transmitter and transmission channel for receiver inputs. Each of the blocks shown in Fig. 1 will be designated as programs or program blocks.

In connection with receiver operation, the following nomenclature will be used in this report. The message bits are convolutionally encoded into channel bits to introduce controlled redundancy for later error control. The channel bits are used to biphase modulate (multiply by ±1) channel symbols. Each channel symbol is made up of a sequence of signaling intervals called chips. In binary minimum-shift-keying (MSK), each chip consists of a sine wave at one of two frequencies, where the frequencies have the minimum separation for orthogonality over the chip interval, and where phase continuity is maintained across chip boundaries. The chip correlator and the channel symbol demodulator taken together constitute a channel symbol matched filter.

III. CONTROL PROGRAM

All the program blocks and support programs are written in the form of subroutines. The Control Program is the supervisor program which calls these programs in the proper sequence. Among the functions of the Control Program are system timing, initialization of magnetic tape drives, initialization of all programs, changing input parameters, recording of results, and various bookkeeping chores.

A table of program names, called the Program Table, is assembled with the system and determines which programs will be executed by the Control Program. If the execution of a different set of program blocks or even only one program block (for individual program checkout) is desired, it requires a simple change in the list of names in the Program Table.

Even though the simulations must demonstrate that the proposed receiver can meet specifications while operating in real-time, it is unnecessary for the simulated receiver to actually stay in step with real-time. In fact, so much computational time is taken by the Control Program, the statistics-gathering routines, the transmitter, and the channel that real-time operation is impossible on the facility's relatively slow computer. Hence, inputs to the receiver are computed or read from a magnetic tape without reference to the true sampling rate or message rate. This simplification eliminates the need for an external clock and the associated routines which react to an external interrupt for processing an incoming sample immediately on arrival. In practice, implementation of these functions is straightforward. Instead, the Control Program's internal clock consists of a counter set to count any convenient quantity which reflects the passage

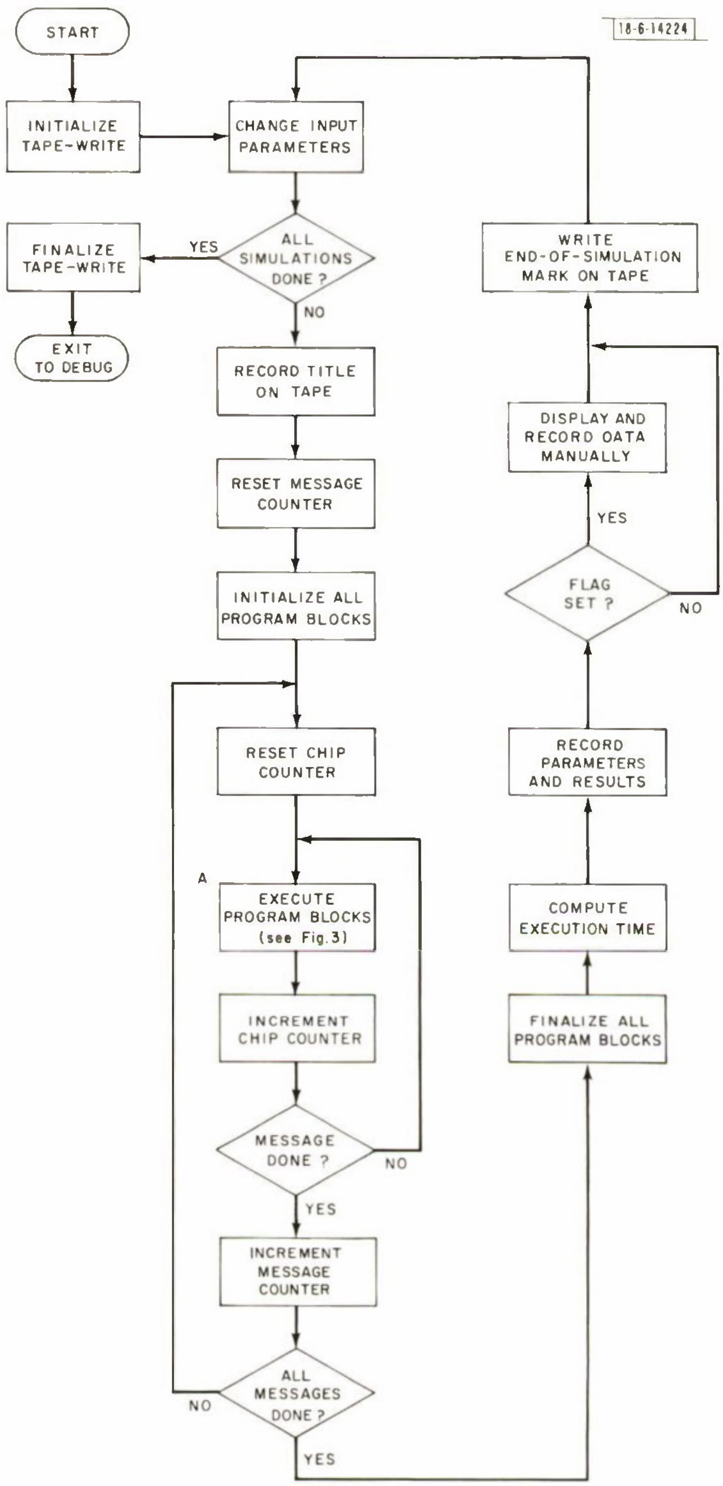

Fig. 2. Flow chart of Control Program.

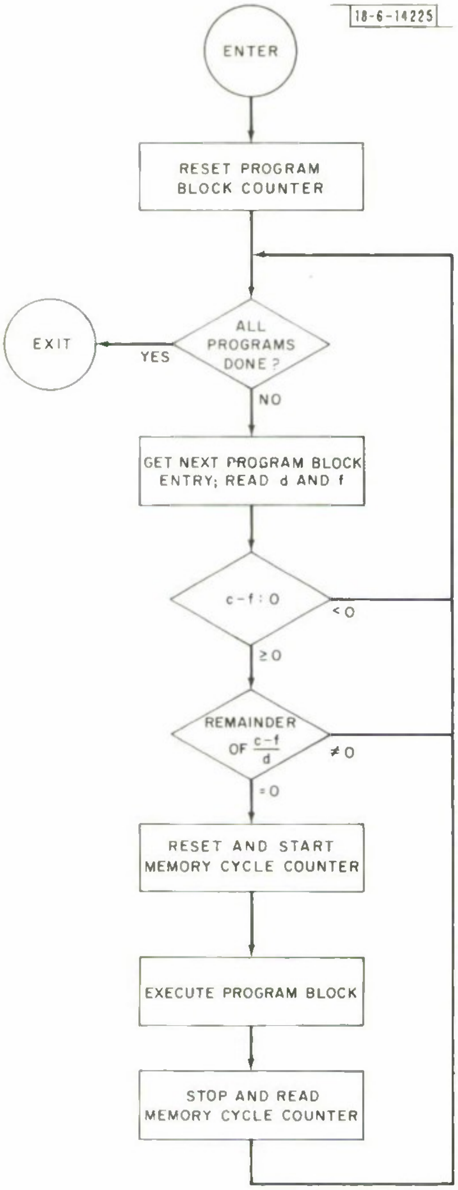

Fig. 3. Flow chart of block A in Fig. 2, where c = chip counter value, f = number of chips delay before a block is first called, and d = number of chips delay between calls.

of time. For instance, in the Sanguine system it counts chips; these were chosen rather than samples or channel symbols in order to achieve a good balance between overhead computational time in the Control Program and memory storage required. All program blocks are called at intervals specified by the number of chips that have been processed. The Program Table includes timing information indicating how much delay occurs before each program is first called and how much delay occurs between calls. In this way, all programs, including those that operate once per chip, once per channel symbol, and once per message, can be accommodated. Since no program block may be called more often than once per chip, each of the filter programs must generate a buffer containing a chip's worth of samples each time when called. For an accurate measure of real-time performance, the Control Program records the number of memory cycles taken by the program blocks.

Although extensive operator interaction with the system is possible when desired, the system is also capable of running without operator intervention. Input parameters can be specified so that the system can run continuously overnight or over weekends. A flow chart of the Control Program is illustrated in Figs. 2 and 3. The Control Program begins a simulation by initializing parameters and all program blocks. Next, the Control Program begins execution of each of the program blocks in order; some programs are executed every chip while others are executed less frequently. Once a message has been processed, the Control Program increments the message counter, resets the chip counter to the beginning of a message, and begins processing the next message with different noise inputs. When the specified number of messages has been processed, the Control Program finalizes the program blocks, computes the execution time for the receiver, and records input parameters and intermediate and final results on magnetic tape. The Control Program then begins a new simulation with a different set of parameters. During this process, data may be manually recorded on tape or plotted on the CRT.

IV. EXPERIMENTAL PROCEDURE

The entire simulation system is designed to be convenient and flexible to use so that modifications in receiver design are easily implemented and analyzed. To facilitate user interaction with the system, the Control Program operates in what is called the DEBUG environment. DEBUG is a program which allows the user to:

(a) Enter and alter parameter values in various formats,

(b) Load and dump programs or data,

(c) Test and correct programs,

(d) Record and display results.

In the application of these operations, the convenience of mnemonic identification of data is available through use of the Item Table. A parameter in the Item Table is described by a symbolic name (up to four characters). The user may refer to the core locations for that parameter by its symbolic name rather than its numerical address. For instance, if FILT is the symbolic name associated with the memory address of the output buffer of a filter program, the user may access the contents of that buffer for printing on the teletype, plotting on the CRT, or recording on magnetic tape simply by typing @FILT/, @FILT$, or @FILT#, respectively, on the teletype. This example indicates the ease with which program block outputs can be examined during the process of debugging programs. When a set of parameters is inserted into the system, the identification of parameters with symbolic names and appropriate format is a tremendous advantage in facilitating insertion and reducing errors.

Each program block is initially tested, debugged, and optimized individually using the simulation system. (In order to isolate a single program for checkout, the Program Table contains only the one entry referring to that program.) For debugging purposes, frequent use is made of (a) Fourier Transform routines to take the transform of block outputs, (b) plotting routines to plot block outputs and their transform, and (c) statistics routines to calculate the mean, standard deviation, and amplitude probability density of block outputs. After each block has been optimized individually as well as can be done without interaction with the rest of the system, the programs are assembled together into larger units for further optimization.

The message processing rate through the entire simulated Sanguine transmitter, channel, and receiver is limited by the fact that in order to adequately represent an analog waveform in a digital manner, sampling rates must be higher than twice the highest frequency of interest. For ELF noise waveforms the sampling interval must be of the order of milliseconds, which limits the processing rate to a few messages per night. At this rate, it would take years to process enough messages through the entire receiver to determine whether requirements are met. However, it is possible to achieve the same results more economically by taking advantage of the fact the Sanguine receiver matched filter outputs can be represented as Gaussian random noise having a mean proportional to the signal level.[2] Thus, optimization of the Sanguine receiver can be separated into two parts. The first part consists of maximizing the signal-to-noise ratio at the output of the matched filter when using MSK signals and ELF atmospheric noise as receiver inputs. The second part consists of optimizing the AGC, phase tracker, and sequential decoder when using intended channel symbols and zero mean Gaussian noise as inputs to the AGC (Fig. 4). In this case, the signal generator block generates the voltages ($\pm V$) associated with the channel symbols while the channel block simulates the amplitude and phase distortion in the generated signal due to natural or man-made changes in the path (i.e., the outputs are $A \cos(\Theta)$ and $A \sin(\Theta)$, where Θ is the phase angle and A is proportional to the channel symbol voltages). All other blocks are described in Section VI or Ref. 3.

The separation of tasks permits a significant savings in computer processing time. In the first part, only a few messages need be processed for good statistical results, while in the

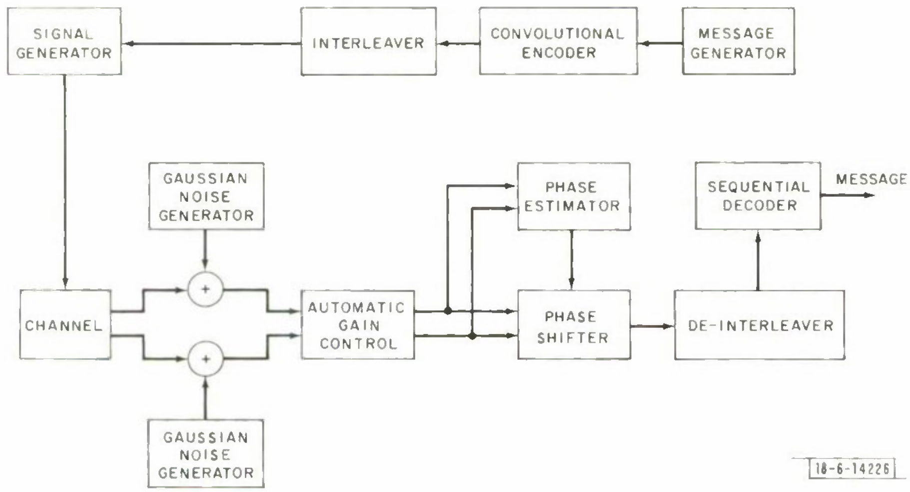

Fig. 4. Block diagram of system for optimizing AGC, phase estimator, and sequential decoder.

second part many messages can be processed with the receiver operating almost three orders of magnitude faster when all the filters are bypassed. Only by using this separation is it feasible to process the large numbers of messages necessary to verify whether error and missed message requirements have been met. Even so, an overnight run is still required to make a signal-to-noise measurement or to determine a probability of missed message.

In order to maintain continuous computer operation, the daytime is normally reserved for program writing and debugging and short simulations, while the nighttime and weekends are reserved for unattended operation consisting of several simulations with different input parameters.

V. SUPPORT SOFTWARE

All the software is written in assembly language with fixed-point calculations to minimize storage required by programs and processing time. The computer operating routines such as the DEBUG operating environment, tape-handling routines, text editor, and assembler were written at Lincoln Laboratory. In addition to these general-purpose operating routines, the following support programs were developed specifically for the simulation facility.

A. Pseudo-Random Bit Generator

The pseudo-random bit generator is a maximal length sequence generator described in Ref. 6. Stored in memory are a mask and a 16-bit integer. The pseudo-random bit is the parity of the masked 16-bit integer. A new integer is then obtained from the old integer by logical shifting left once and adding the pseudo-random bit. This procedure is illustrated in Fig. 5.

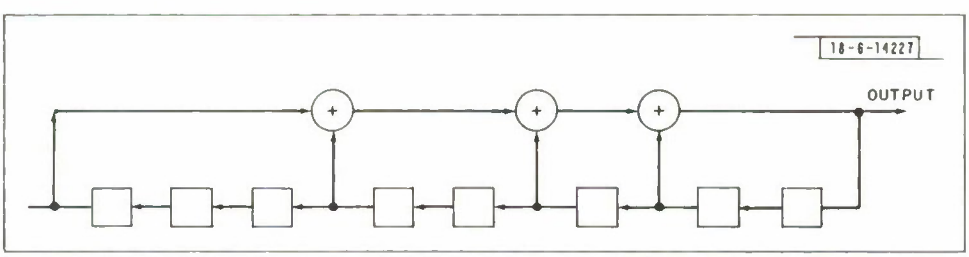

Fig. 5. Illustration of pseudo-random bit generator algorithm. In this illustration the mask is 226_8 and the integer is 8 bits long.

B. Uniform Random-Number Generator

Since the random-number generator can have profound influence on the simulation results, considerable care was taken to select one which is fast and convenient and yet to ensure that the random numbers have the necessary properties and that the period of the generator is much larger than the number of samples used in a single simulation. There are two random-number generators in the simulation system, both of which generate linear congruential sequences.[7]

The generator which produces random numbers with the best characteristics is based on the following algorithm:[8]

$$x_n = 11^5 x_{n-1} \bmod (2^{31} - 1) \qquad \text{period } 2^{31} - 2.$$

The resulting sequence of integers takes each value in the closed interval 1 to $2^{31} - 2$ just once before recommencing its cycle. This arises because $2^{31} - 1$ is an Euler prime and 11^5 is a primitive root of this prime. Although this generator yields good results, it is necessary to implement the algorithm with double precision calculations.

The second generator in the system is based on the following algorithm:

$$y_n = (a\,y_{n-1} + b) \bmod 2^{15} \qquad \text{period } 2^{15}$$

$$z_n = \begin{cases} (c\,z_{n-1} + d) \bmod 2^{15} \\ d \text{ if above } 0 \end{cases} \qquad \text{period } 2^{15} - 1$$

$$x_n = y_n \oplus z_n \qquad \text{period } 2^{15}\,(2^{15} - 1),$$

where $\oplus$ is the exclusive-OR operation. The coefficients, listed in Fig. 6, were tested so that the individual y_n and z_n have desirable properties. Since all calculations are performed single precision, this generator is much faster than the first generator.

18-6-14228

a	b	c	d
14641	127	15385	517
15625	3	30585	181
22637	5801	12589	7187

Fig. 6. List of random-number generator coefficients.

Testing of random-number generators is a non-trivial problem[7] and is too lengthy to be discussed here. Extensive tests were made on the first generator and yielded good results; it is used in the simulation system where randomness is critical. For instance, it is used to generate noise samples for simulated ELF atmospheric background noise, and to generate Gaussian channel symbol noise when all the receiver filters are bypassed (Section IV). Extensive tests on the second generator yielded locally good results; it is used where randomness is less critical, such as when only a few random numbers are needed in a simulation run. For instance, it is used to generate random receiver antenna depths to simulate a moving platform, and to generate random signal phases to simulate random phase shifts in the atmospheric channel.

C. Gaussian Random-Number Generator

The Gaussian random-number generator applies a table look-up method to map a uniform random number into a Gaussian random number.[9] This method is both fast and accurate.

D. Fast Fourier Transform

The discrete Fourier Transform of a set of 2^k complex numbers is calculated with a fast Fourier Transform (FFT) algorithm using radix 2 and a decimation in time method, which permits the transform to be done in place without requiring extra data storage.[10] This routine requires samples of only a quarter cycle of a sine wave for its reference table. To reduce running time, the reference table is generated once at the beginning of a run and not every time a transform is performed. The arithmetic used in the computations is block floating-point. (Block floating-point arithmetic is the same as fixed-point arithmetic with a scale factor of a power of two kept in an auxiliary register.) In this way, the full 16-bit accuracy of the computer is utilized while providing a large dynamic range. This technique also prevents overflows during the computation of the transform. The time in seconds required to do a complex valued N point transform, where $N = 2^k$, is

$$T_{FFT} \approx 2 \times 10^{-4} \, N \log_2 N \text{ sec}$$

The running time for a 4096 point complex transform is ~10 seconds.

The FFT routine is used for two main purposes in the simulation system: (1) spectral analysis of noise, and (2) frequency spectra of filter program outputs as a debugging aid. While the FFT routine could be used to replace a cascade of filters, this FFT routine is too slow for real-time processing of signals. It is faster in the simulation system to implement the filters as recursive difference equations.

E. Statistics

There are two statistics routines in the simulation system. The first calculates the mean and standard deviation of a set of data. The second calculates the amplitude probability density of a set of data for specified bin widths and bin limits.

F. Single-Precision Functions

The following fixed-point single-precision functions are available in the system:

sine	arctangent (full quadrant)
cosine	square root
arctangent (principal value)	logarithm

The sine and cosine routines use a table look-up with linear interpolation in order to save time. The square root routine uses the Newton-Ralphson formula. The arctangent and logarithm routines use a seventh degree and fifth degree Chebyshev polynomial, respectively. The logarithm, the square root, and the principal valued arctangent routines were supplied with the computer. Other function routines are available but have not been incorporated into the simulation system.

G. Double-Precision Routines

The following double-precision routines are available in the system:

double precision add	double precision complement
double precision subtract	double precision square root (with single precision result)
double precision multiply	
double precision divide	

It should be noted that many program blocks perform double precision calculations without using the above routines in order to save computational time at the expense of a slightly more complicated computer program. The first five routines were supplied with the computer.

H. Magnetic Tape-Handling Routines

There are three executive tape-handling routines in the simulation system: a tape-read routine, a tape-write routine, and a tape-translator routine. Their purpose is to free the user from concerning himself with most of the details associated with the input and output of data on magnetic tape, such as:

(1) Tape positioning to load point or any other point,
(2) Buffering on either input or output,
(3) Identifying data with labels, numerical modes, and buffer sizes,
(4) Searching for a particular item of data,
(5) Tape error checking,
(6) Writing logical end-of-file marks on the output tape at the conclusion of a run.

A primary virtue of all the tape-handling routines is that they are easy and convenient to use. In this way, the user is encouraged to record all pertinent information concerning a simulation run. For example, in order to record a block of data associated with a particular parameter which is identified with the symbolic name ITEM in the Item Table, the user (1) inserts the instruction

```
CALL      $RCD,'ITEM'
```

into his program for recording under program control, or (2) types @ITEM# on the I/O terminal to initiate recording. Maximum throughput rate is needed especially when the read routine is used for the input of atmospheric noise samples or signal plus noise samples recorded previously on tape. To minimize the time absorbed by data transfers, the read routine is double buffered and utilizes block data transfer. On the other hand, maximum throughput rate is not needed for recording simulation output data consisting mainly of input parameters and performance statistics. Hence, the write routine is single buffered. When the end-of-tape mark is reached on the output tape, the write routine can switch to a new output tape on a different tape transport. This is a useful feature when the simulation runs unattended over a long period of time, such as over a long weekend.

The translator routine is used to copy selected information from tapes written by the write routine onto tapes in a format suitable for listing on high-speed printers available elsewhere in the Laboratory or for listing on the CRT terminal. Results can be analyzed from the printed listing or can be further processed from the output tape. Figure 7 shows an example of a listing. Identification of output data with a four-character mnemonic name is a particularly helpful feature of the simulation system when analyzing results. A title and assembly date are printed at the top of every page. In this particular example, the following identifications are made:

DATE - month and day on which the run was made (date shown is Dec. 24),
DEPT - receiver antenna depth in meters,
CUTO - receiver bandwidth,
FREQ - signal center frequency, signal deviation frequency, power line frequency, notch filter pole and zero frequency,

P-MS - mean and standard deviation of received signal phase in radians,

L-ST - channel symbol statistics both subsequent to and prior to the phase estimator. (The fourth column is the mean in single precision and the seventh and eighth column is the variance in double precision.)

S/N - signal-to-noise ratios in dB subsequent to and prior to the phase estimator.

```
    1  WISCONSIN TRANSMISS:ON                DEC 18  1971                  -6-14223

OATE
   12.24

OEPT
   100.

CUTO
   200.

FREO
   72.00          1.602          60.00          60.00          60.00

P-MS
     .7511          1757

L-ST
   1058.      1.     780.   3019.    5671.     0.     549.   15718.    0.   42
   1059.      1.     780.     60.   19156.     0.     495.   10641.    0.   40
   7052.    780.     780.   2153.    9284.     0.     550.   15528.    0.   42
   7832.    780.     780    2083    21005.     0.     498.   23633.    0.   40

S/N
   -2.957         -5.902
```

Fig. 7. Example of a listing with typical parameters from a single simulation.

I. Plotting Programs

The plotting programs facilitate the display of data in graphical form on the CRT terminal. These routines are designed to eliminate as much as possible the tedious chores associated with graphing and still retain enough flexibility to accommodate a wide variety of graphs. With these routines the user need not concern himself with specific commands directed to the CRT; the programs will automatically:

(1) Scale axes,

(2) Position axes on the CRT screen,

(3) Place either grid marks or grid lines on axes,

(4) Label axes and title graphs,

(5) Number axes limits,

(6) Perform linear transformation of data coordinates to screen coordinates,

(7) Plot data curves on grids.

The versatility of these routines is indicated by the options available to the user:

(1) One set of axes may be displayed full screen or two sets of axes may be displayed with one in the top half and the other in the bottom half for direct comparison.

(2) Axes scaling may be specified or determined by the limits of data to be displayed.

(3) Data points may be placed on the axes with or without lines joining the points.

(4) Several sets of data may be displayed on the same axes, each set having points marked with a different character.

(5) The x coordinates of data may be specified explicitly or implicitly in the form of an initial value and uniform increments along the x axis.

(6) The point coordinates may be stored in a buffer and plotted consecutively or may be individually added to a set already displayed on the screen.

In the simulation system, extensive use is made of an executive plotting routine which sacrifices this versatility for ease and convenience to the user. With this routine, only one set of y coordinates stored in a buffer may be plotted against a buffer index value on a set of axes drawn full screen, where the y axis limits are the extrema of the data values. For example, to graph a buffer of data associated with a particular parameter identified with the symbolic name ITEM in the Item Table, the user need only type @ITEM$ on the I/O terminal. Thus, this routine permits rapid examination of buffer contents, which is an extremely useful aid in debugging programs. (Figures 13 through 16 are examples of graphs made on the CRT with the plotting programs but redrawn for this report.)

VI. SIMULATED SANGUINE TRANSMITTER AND TRANSMISSION CHANNEL

In order to operate the simulated Sanguine receiver,[1,3] a simulated transmitter and transmission channel are necessary. As with the support software, these programs are written in assembly language with fixed-point calculations throughout to minimize processing time and storage required. For convenience in the simulations, each program has its own output buffer. Since the Control Program calls each program at most once per chip, those programs which output signal and noise samples must store all the samples for that chip in a buffer.

Due to the 16-bit word size of the computer, significant effort is often required to perform simple calculations in single precision and still maintain accuracy. Nevertheless, double precision calculations are avoided in order to save processing time unless receiver performance is thereby impaired. In those programs where it is not clear whether single precision is adequate, double precision is used; further simulations can indicate whether this is necessary. Although calculations may be performed double precision, the output from each program is single precision. The savings in processing time and space required is significant, particularly for those programs which manipulate signal and noise samples.

A. Message Generator

Simulated messages are segments of a sequence of bits generated by the pseudo-random bit generator. The mask of the bit generator is 70610_8 for which the period of the message generator is $2^{15} - 1$ bits. When the initial integer or the mask is zero, the all 0 message is generated.

B. Convolutional Encoder

The message is convolutionally encoded to introduce controlled redundancy for error correction. Encoding of message bits into channel bits is explained in Refs. 11 and 12. If the message is all 0's, the channel bits are all 0's, and the signal-to-noise ratio of the channel symbols after the matched filter can be calculated easily as the mean squared divided by the variance of the channel symbols.

C. Interleaver

The channel bits are interleaved over the entire message in order to minimize the effects of burst noise, channel anomalies, and correlated phase errors and to shorten message delivery times when signal-to-noise ratios are high.

D. Chip Encoder

In order to achieve band spreading, the chip encoder encodes each channel bit into a specified number of chip commands; each chip command is the mod 2 sum of the channel bit and a keystream bit with 0 and 1 mapped into −1 and +1, respectively. The sequence of chip commands is then used in the waveform generator to determine signaling frequency. The keystream consists of a pseudo-random bit generator having a mask of 141554_8 for which the period is $2^{16} - 1$ bits. The periods of the message generator and keystream are non-commensurate, thereby minimizing correlation.

E. Waveform Generator

The Sanguine signals are binary MSK waveforms in order to make efficient use of the bandwidth and present minimal transients to the transmitter. Binary MSK waveforms are phase continuous FSK waveforms with the frequency deviation chosen to be the minimum consistent with orthogonality between waveforms over the signal duration.[13] Let

T_c = chip duration in seconds,

f_c = center frequency,

f_d = deviation from center frequency.

Binary MSK generation occurs only when $4f_cT_c$ is an integer and $4f_dT_c = 1$.

The waveform generator generates MSK waveforms limited to 2-phase switching;[13] i.e., $4f_cT_c$ is an odd integer. These waveforms can be represented by the following formula:

$$y(t) = Apa_{t-T_c} \cos[\omega_c \bar{t} + a_{t-T_c} a_t \omega_d \bar{t} + \theta] \quad , \qquad (1)$$

where

t = time measured from the start of the first chip,

$\bar{t} = t \bmod T_c$,

$y(t)$ = signal value at time t,

A = signal amplitude,

θ = phase angle,

$\omega_c = 2\pi f_c$,

$\omega_d = 2\pi f_d$,

a_t = chip command (at time t) = ± 1,

$$p = \begin{cases} 1 & \text{if } 3f_cT_c \bmod 4 = 3 \\ \begin{bmatrix} 1 & \text{if chip number even} \\ -1 & \text{if chip number odd} \end{bmatrix} & \text{if } 4f_cT_c \bmod 4 = 1. \end{cases}$$

Note that the signaling frequency, which is constant over a chip's duration, is determined by the product of the present and the previous chip command. When $4f_c T_c \bmod 4 = 3$, there is an integral number of cycles at the upper frequency over a chip's duration. When $4f_c T_c \bmod 4 = 1$, there is an integral number of cycles at the lower frequency over a chip's duration, and p is the parity of the chip number. When $\theta = 0$, chip boundaries occur at signal peaks. When $\theta = \pi/2$, chip boundaries occur at zero signal. Through the use of $\bar{t}$ in the above formula, time is mapped into the interval 0, T_c. The purpose of this mapping is that signals may be calculated for each chip without reference to the signal phase of the previous chip. Cumulative quantization errors are thereby eliminated.

To implement the waveform generator on the digital computer, the signal y(t) is sampled a specified number of times per chip. With each execution of the signal generator, all the samples for that chip are stored into a buffer. For convenience, the sampling rate is chosen so that a chip contains an integral number of samples.

In order to maintain large dynamic range on the atmospheric noise and still keep signal plus noise single precision, signal levels must be quite low, ~3 to 10 digital units. Unless precautions are taken, the quantization errors introduced with this signal level can be significant. The procedure used in the simulated signal generator to reduce these errors is the following. The signal amplitudes are stored with a scale factor of 2^7, while the cosine function in Eq. (1) is calculated with a scale factor 2^8. With these scale factors, and a 16-bit computer, each may be specified to better than 1 percent accuracy. Their product is scaled by 2^{15}, so that the high order term of the product has the appropriate scale 2^0. A uniformly distributed random number ϵ, $0 \leqslant \epsilon < 1$, which differs for each sample, is then added to the result. To see the effect of ϵ, consider the separation of y(t) into its integer part and its fractional part,

$$y = I + f \quad ,$$

where $0 \leqslant f < 1$. Let square brackets around an expression designate the integer part of the expression; i.e.,

$$[y] = I \quad .$$

Then

$$\begin{aligned} [y + \epsilon] &= I & &\text{if } 0 \leqslant \epsilon < 1 - f \\ &= I + 1 & &\text{if } 1 - f \leqslant \epsilon < 1 \quad . \end{aligned}$$

The average of this expression over all ϵ is

$$\begin{aligned} \overline{[y + \epsilon]} &= (1 - f)\, I + f(I + 1) \\ &= I + f \quad . \end{aligned}$$

Thus, although any individual calculation of y(t) can have a quantization error of up to one digital unit, which is a major fraction of the signal level, the quantization error is eliminated from the average value of y(t). The noise power introduced by ϵ will have negligible effect at the output of the matched filter.

F. Atmospheric Channel

Amplitude and phase distortion in the generated signal due to natural or man-made changes in the atmospheric channel can be simulated by changing A and θ [Eq. (1)] as a function of time.

For example, the transition from daytime to nighttime is simulated by a slowly varying signal amplitude and signal phase. A nuclear detonation in the atmospheric channel is simulated by a decreased signal amplitude accompanied with an abrupt change in the signal phase followed with a rapid change in the signal phase. Other variations are possible, such as sinusoidal or random changes in either amplitude and phase or both.

G. Atmospheric Noise Source

The purpose of the atmospheric noise source is to provide noise samples which are added to the signal. These noise samples may be one of three types:

(1) Noise recorded on tape,

(2) Simulated ELF atmospheric noise,

(3) Gaussian random noise.

Atmospheric noise has been recorded wideband (7 to 350 Hz) at several sites around the world, including Massachusetts, Florida, Saipan, Malta, and Norway. These data can be used (after converting to a format which can be read by the simulation read routine) as input to the simulation system. However, each tape is limited by the length of tape to a half-hour segment, which precludes its use for long simulations. The noise characteristics and processing results with these tapes are discussed in Ref. 2.

In order to provide extended amounts of data for ELF simulations and to provide a reference for calculating noise processing performance of various nonlinear processing, an ELF noise model was developed[4] and serves as the second type of noise input.

Independent Gaussian random noise, the third type of noise input, can be added to the signals to provide known characteristics for checking receiver programs.

H. Man-Made Interference Generator

A sine wave of variable amplitude and frequency can be added to the transmitted signal plus noise. This sinusoid can be used to simulate power line interference at 50 or 60 Hz or a simple sine wave jammer.

I. Ocean Channel

The transmission coefficients between the atmospheric channel and the ocean channel are given by

$$\frac{H(z = 0^+)}{H(z = 0^-)} = 1 \quad ,$$

$$\frac{E(z = 0^+)}{H(z = 0^-)} = \eta = (1 + j)\sqrt{\frac{\pi f \mu}{\sigma}} \quad ,$$

where η is the wave impedance relating E- and H-fields in the ocean, f is the frequency of an incident sinusoid, z is the coordinate measuring depth into the ocean, σ is the electrical conductivity of the ocean, μ is the permeability, $j = \sqrt{-1}$, and a smooth-surfaced interface is assumed. Propagation through the ocean channel can be described by the frequency response function W(f, d) giving the complex amplitude of a sinusoid as it progresses to a depth z = d.

$$W(f, d) = e^{-(1+j)d/\delta} \quad ,$$

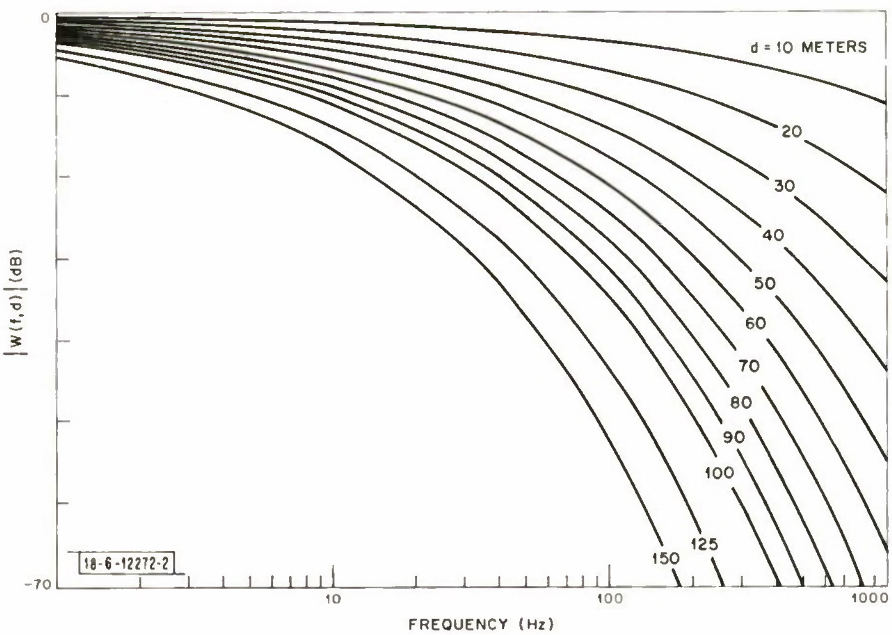

Fig. 8. |W(f,d)| vs frequency.

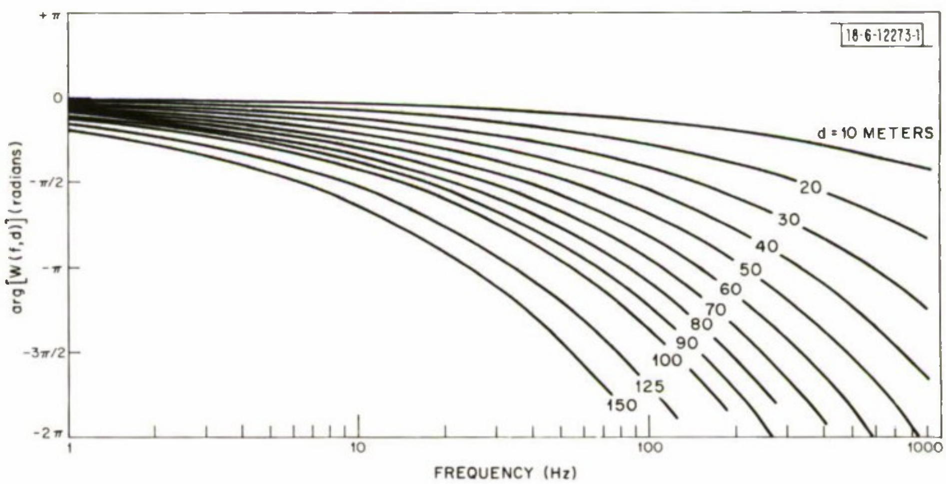

Fig. 9. Phase shift, arg [W(f,d)], vs frequency.

where the skin depth δ is given by

$$\delta = \frac{1}{\sqrt{\pi f \mu \sigma}} \quad .$$

W(f, d), which is the same for either E- or H-field propagation, can be represented by a low-pass filter. Figures 8 and 9 plot the magnitude and phase of W(f, d) for various depths and for σ = 4 mhos/m.

The receiving antennas under investigation are electrode pair antennas, which sense the E-field, and loop antennas, which sense the dH/dt field. The ocean channel with H-field as the input and horizontal antenna deployment exhibits a bandpass characteristic as can be seen from a plot of

$$W_H(f, d) = j2\pi f W(f, d)$$

for the dH/dt field channel response (Fig. 10), or

$$W_E(f, d) = \eta W(f, d)$$

for the E-field channel response (Fig. 11). Both an amplitude attenuation and a phase shift are imparted to the signal by the ocean channel.

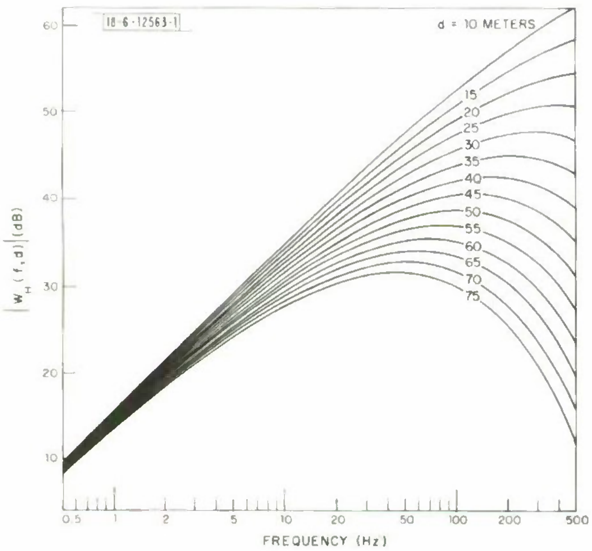

Fig. 10. $|W_H(f,d)|$ vs frequency.

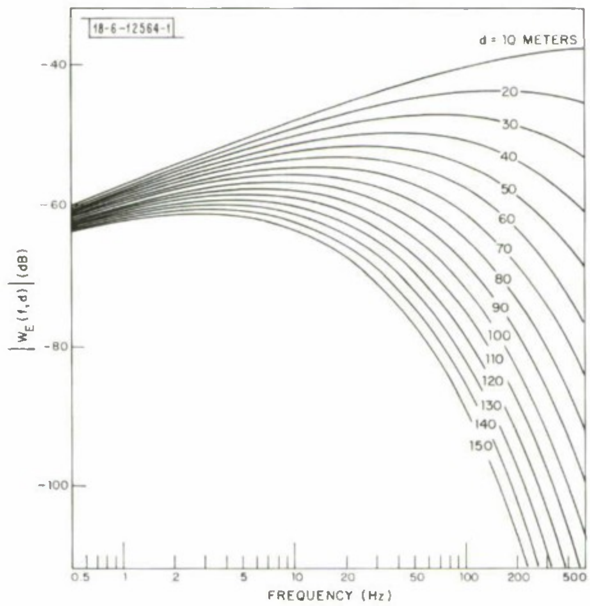

Fig. 11. $|W_E(f,d)|$ vs frequency.

Signals at depth can be obtained from signals at the surface by convolving with the ocean channel frequency response function. However, without special purpose hardware it is faster and more convenient to solve the set of partial differential equations that characterize the fields at depth z in the ocean:

$$\frac{\partial H}{\partial z} = \sigma E \quad ,$$

$$\frac{\partial E}{\partial z} = -\mu \frac{\partial H}{\partial t} \quad ,$$

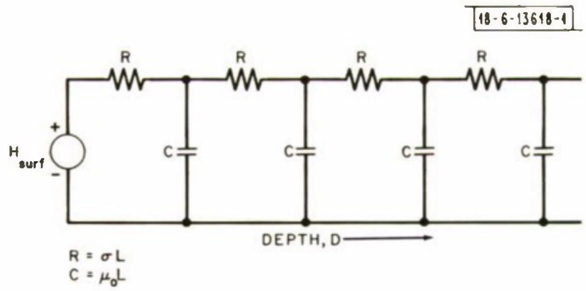

Fig. 12. Transmission line representation of ocean channel.

where the boundary condition is given by the continuity of the H-field at the surface. The numerical technique for solving these equations[14] utilizes both time and spatial quantization to transform differential equations into difference equations over a net of space-time coordinates $\{z_k, t_n\}$. Since the differential equations are the same as those describing a lossy RC transmission line, the spatial quantization is analogous to the representation of a transmission line by a cascade of RC networks (Fig. 12). For the transmission line, let r be the resistance per unit length, c be the capacitance per unit length, and L and T_{sample} be the quantization units of length and time, respectively. A difference equation for approximating the voltage on the k^{th} node can be written as

$$V_k(n) = V_k(n-1) + \frac{T_{sample}}{CR} \{V_{k-1}(n-1) - 2V_k(n-1) + V_{k+1}(n-1)\} \quad ,$$

where R = Lr and C = Lc (Ref. 14). The solution to this equation is stable when

$$\frac{T_{sample}}{CR} = \frac{T_{sample}}{L^2 cr} \leqslant \frac{1}{2}$$

The fields in the ocean channel can be obtained from this equation provided that the RC network is driven by a voltage source proportional to the H-field at the surface and the following transformations are made:

$$I \rightarrow E$$

$$V \rightarrow H$$

$$r \rightarrow \sigma$$

$$c \rightarrow \mu$$

Each section represents an increase in depth of L meters in the ocean. The currents through the resistors are approximations to the E-field at depths 0.5L, 1.5L, 2.5L, etc. The stability condition on the field calculations is

$$\frac{T_{sample}}{L^2 \mu\sigma} \leqslant \frac{1}{2} \quad ,$$

from which a lower limit on L may be determined when T_{sample} is chosen as the reciprocal of the sampling rate.

For the simulation, a finite number of sections are taken and the resulting network is solved by recomputing the fields after every sampling period. Block floating-point calculations are used to great advantage for preserving accuracy, since the ocean channel input has a broad dynamic range. It has been found experimentally that to simulate accurately the effects of the ocean channel at depths of interest (1) only eight sections of line are needed, and (2) the E-field is calculated between the 2nd and 3rd nodes for $d < 120$ m and between the 3rd and 4th nodes for $d \geqslant 120$ m. Figure 13 shows the frequency response of the simulated ocean and the theoretical response of an ideal ocean model for an antenna depth of 100 m, σ = 4 mho/m, and T_{sample} = 1 msec. The

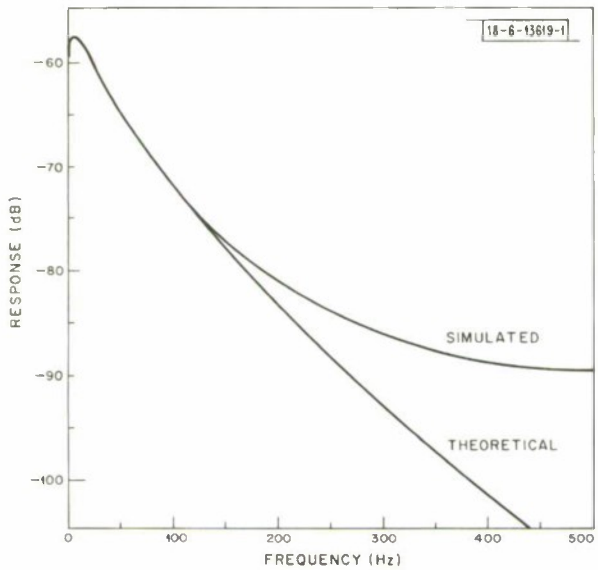

Fig. 13. Simulated and theoretical frequency responses of ocean with d = 100 m.

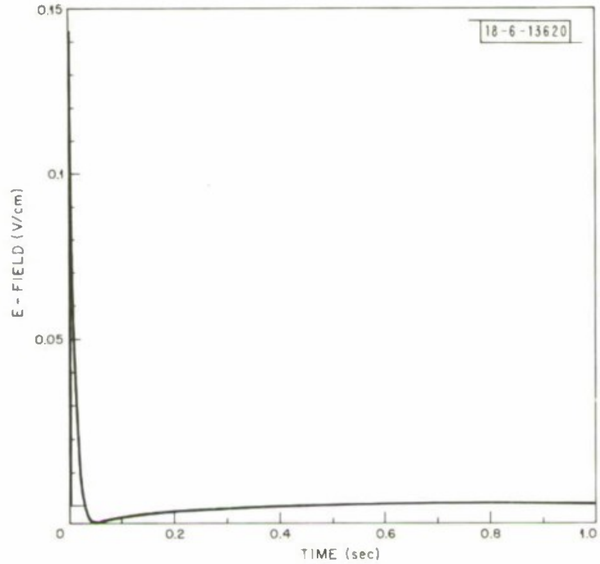

Fig. 14. Simulated impulse response of ocean with d = 100 m.

observed deviation at the higher frequencies is not significant because (1) it does not correspond to a great error in depth (at this depth it is only 10 m at 350 Hz, for example), and (2) the filtering prior to the nonlinearity will include a low-pass filter that will cut off at frequencies above where the deviations are measurable. Figure 14 shows the time response of the simulation ocean (E-field) due to an impulse (H-field) at the surface. The theoretical impulse response is given by

$$e(t) = \sqrt{\frac{\mu_o}{4\pi\sigma}} \left(\frac{d^2\mu\sigma}{2t} - 1\right) t^{-3/2} \exp\left[-\frac{d^2\mu\sigma}{4t}\right] \quad .$$

The simulated and theoretical impulse responses exhibit only minor differences. Figures 15 and 16 show recorded atmospheric noise before and after passing through the simulated ocean

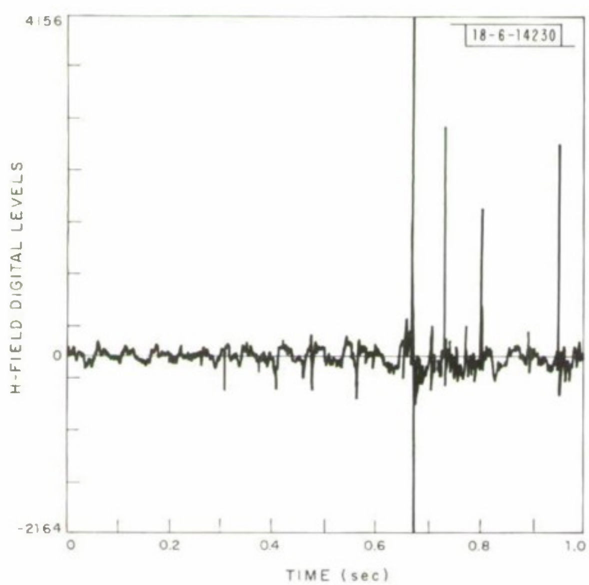

Fig. 15. H-field at surface.

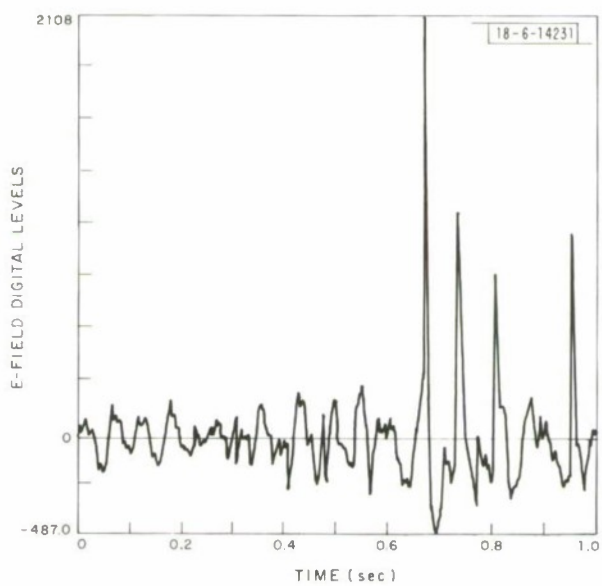

Fig. 16. E-field at d = 100 m.

channel to a depth of 100 m. Note that the major effect of the ocean channel is to smear the noise pulses; i.e., they are reduced considerably in magnitude with respect to the background level and extend over a much longer interval.

J. Depth Generator

To simulate a moving receiver platform, the depth generator provides variations in the depths at which fields are calculated in the ocean channel. Depths may vary linearly, sinusoidally, randomly, or in a stepwise fashion.

K. Antenna Noise Source

Antenna noise samples can be added to the signals at depth to simulate the effect of the antenna. Currently, the noise samples are independent Gaussian random noise, but can be changed, for example, through linear filtering. Processing in the presence of antenna noise is described in Ref. 2.

APPENDIX A
HARDWARE

The simulation facility has been designed around a Varian 620/i digital computer dedicated to the Sanguine simulation project. The computer has 32,768 16-bit words of memory, a memory cycle time of 1.8 μsec, one accumulator, and two index registers. Included with the computer are a hardware multiply (18 μsec), a direct-memory-access feature, a "buffer interlace controller" (to permit block data transfer between memory and I/O devices), and a 12-bit analog-to-digital converter. The following peripherals are also part of the facility:

Teletype I/O terminal (Model KSR 35 – 10 cps)
Computek CRT graphic display I/O terminal
Tektronix hard copy unit (for copies of CRT displays)
4 PEC 9-track magnetic tape drives (25 ips, 800 bpi)
Remex paper tape reader (300 cps)
Tally paper tape perforator (60 cps)
Real-time clock (counts memory cycles primarily for timing programs).

Figure 17 is a photograph of the facility.

-6-12560

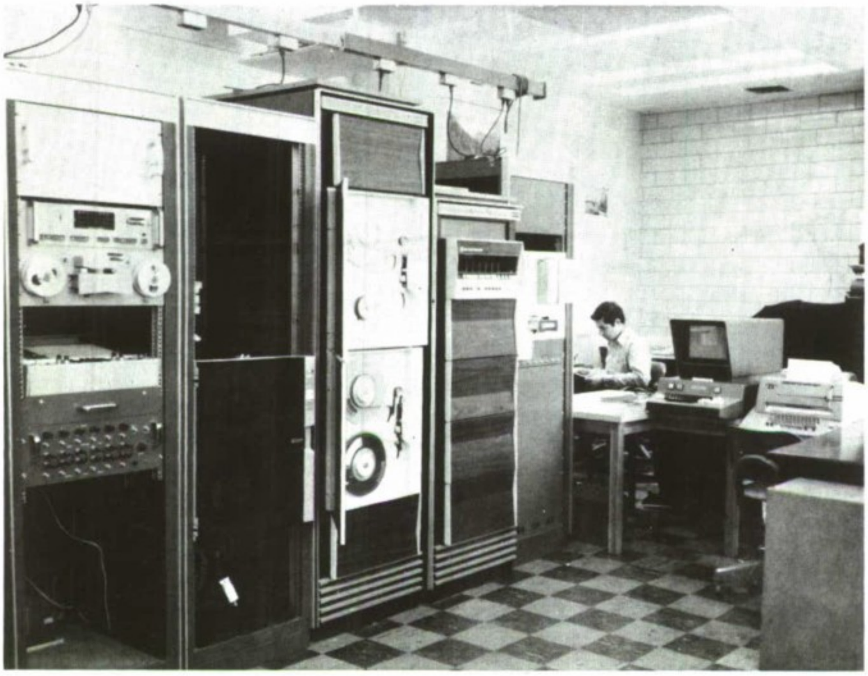

Fig. 17. Computer facility for Sanguine receiver simulation.

REFERENCES

1. S. L. Bernstein, "A Sanguine Receiver Structure," Technical Report 488, Lincoln Laboratory, M.I.T. (limited distribution).
2. A.S. Griffiths, "ELF Noise Processing," Technical Report 490, Lincoln Laboratory, M.I.T. (13 January 1972).
3. S. L. Bernstein, A.S. Griffiths, D. A. McNeill and I. Richer, "An Experimental Sanguine Receiver," Technical Report 492, Lincoln Laboratory, M.I.T. (limited distribution).
4. J.W. Modestino, "A Model for ELF Noise," Technical Report 493, Lincoln Laboratory, M.I.T. (16 December 1971).
5. J.W. Modestino, "A Theoretical Optimum ELF Receiver Structure," Technical Report 494, Lincoln Laboratory, M.I.T.
6. W.W. Petersen, Error-Correcting Codes (M.I.T. Press, Cambridge, Massachusetts, 1961), Chaps. 6 and 7. An exhaustive list of all irreducible polynomials up to degree $p = 17$ is given in Appendix C. To obtain masks which are usable in the simulation system bit generator, these polynomials must be bit reversed and then the least significant bit ignored.
7. D. E. Knuth, Seminumerical Algorithms: The Art of Computer Programming (Addison-Wesley, Reading, Massachusetts, 1969), Chap. 3.
8. H.C. Peterson and E.W. Pike, private communication.
9. I. Richer, "An Ideal Gaussian Random Number Generator for a Small Computer," to be published.
10. B. Gold and C.M. Rader, Digital Processing of Signals (McGraw-Hill, New York, 1969), Chap. 6.
11. I. Richer, "Sequential Decoding with a Small Digital Computer," Technical Report 491, Lincoln Laboratory, M.I.T. (24 January 1972).
12. J.M. Wozencraft and I.M. Jacobs, Principles of Communication Engineering (Wiley, New York, 1965), Secs. 6.3 and 6.4.
13. B. E. White, "A Derivation of the Spectra of N-ary Orthogonal Continuous-Phase FSK Waveforms for ELF/VLF Communications," IEEE Trans. Commun. Tech. Com-19, 536 (1971). Essentially, 2-phase switching implies that chip boundaries occur only at signal peaks or only at zero crossings.
14. F.B. Hildebrand, Finite-Difference Equations and Simulations (Prentice-Hall, Englewood Cliffs, New Jersey, 1968), Secs. 3.2 and 3.3.

DISTRIBUTION

Chief of Naval Operations
Attn: Capt. Wunderlick (OP94P)
Department of the Navy
Washington, D. C. 20350

Chief of Naval Research
Attn: Dr. T. P. Quinn
Department of the Navy
Washington, D. C. 22217

Computer Sciences Corporation
Systems Division
Attn: D. Blumberg
6565 Arlington Boulevard
Falls Church, VA 22046
(3 copies)

IIT Research Institute
Attn: Dr. D. A. Miller
10 West 35th Street
Chicago, Illinois 60616

Institute for Defense Analyses
Attn: Mr. N. Christofilos
400 Army-Navy Drive
Arlington, VA 22202

Naval Civil Engineering Laboratory
Attn: Mr. J. R. Allgood
Department of the Navy
Port Hueneme, CA 93041

Naval Electronics Laboratory Center
Attn: Mr. R. O. Eastman
Department of the Navy
San Diego, CA 92152

Naval Electronic Systems Command
Attn: Capt. F. L. Brand
PME-117
Department of the Navy
Washington, D. C. 20360

Naval Electronic Systems Command
Attn: Mr. J. E. Don Carlos
PME-117T
Department of the Navy
Washington, D. C. 20360
(2 copies)

Naval Electronic Systems Command
Attn: Cmdr. K. Hartell
PME-117-21A
Department of the Navy
Washington, D. C. 20360
(25 copies)

Naval Electronic Systems Command
Attn: Dr. B. Kruger
PME-117-21A
Department of the Navy
Washington, D. C. 20360

Naval Electronic Systems Center
Attn: Capt. J. V. Peters
PME-117-22
Department of the Navy
Washington, D. C. 20360
(2 copies)

Naval Electronic Systems Center
Attn: Mr. E. Weinberger
PME-117-23
Department of the Navy
Washington, D. C. 20360
(2 copies)

Naval Facilities Engineering Command
Attn: Mr. W. W. Pinkerton
Department of the Navy
Washington, D. C. 20390

Naval Underwater Systems Center
Attn: Mr. J. Merrill
New London Laboratory
New London, CT 06321
(4 copies)

The Defense Documentation Center
Attn: TISLA-1
Cameron Station, Building 5
Alexandria, VA 22314

The MITRE Corporation
Attn: Mr. W. F. Longchamp
Post Office Box 208
Bedford, MA 01730
(2 copies)

DOCUMENT CONTROL DATA – R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* Lincoln Laboratory, M.I.T.	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP None
3. REPORT TITLE A Simulation Facility for Communication Systems	
4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)* Technical Report	
5. AUTHOR(S) *(Last name, first name, initial)* McNeill, Dale A.	

6. REPORT DATE 27 December 1971	7a. TOTAL NO. OF PAGES 28	7b. NO. OF REFS 14

8a. CONTRACT OR GRANT NO. F19628-70-C-0230 b. PROJECT NO. 1508A c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) Technical Report 489 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)* ESD-TR-71-324
10. AVAILABILITY/LIMITATION NOTICES Approved for public release; distribution unlimited.	
11. SUPPLEMENTARY NOTES None	12. SPONSORING MILITARY ACTIVITY Department of the Navy

13. ABSTRACT

A simulation facility, consisting of a small general-purpose computer, associated peripherals, and extensive software, has been developed for analyzing communication systems. Presently, the facility is being used to develop an all digital Sanguine receiver and to gather performance statistics on the receiver. The facility and the simulated Sanguine transmitter and transmission channel necessary for operating the receiver are described.

14. KEY WORDS

computer simulation facility	ELF
communication systems	digital receivers