ARCHITECTURES FOR REAL-TIME DIGITAL CHANNEL SIMULATORS

E.P. O'GRADY
Elec. Engg. Dept.
Univ. of Maryland

Summary. This paper suggests an all-digital, time domain approach for real-time simulation of digital communications channels and proposes four possible implementations of the time-domain approach using standard minicomputers or microprocessors and peripheral random number generators. The time-domain method is based on simulating the digital channel on a bits-in, bits-out basis with bit errors introduced into the bit stream in a manner which approximates the error sequence of a real (or hypothetical) communications channel. The error sequence of the simulator can duplicate a measured error sequence or it can be generated by a stochastic model of the error sequence. The four proposed implementations represent different levels of complexity in the architecture of the channel simulator. The first proposed implementation employs only a single computer; the second employs a computer and a peripheral random number generator; the third employs a computer and multiple peripheral random number generators; the fourth employs multiple computers and multiple peripheral random number generators. The significance of the time-domain approach lies in its potential application to the design of high performance, general-purpose media simulators at greatly reduced cost due to the use of standard hardware and relatively simple processing.

Introduction. This paper suggests an all digital approach to simulating digital communications channels directly in the time domain. The approach is markedly different from the approached used in current channel simulators and takes advantage of the availability of low-cost high-speed minicomputers and microprocessors. A real-time simulator employing this approach introduces errors into a bit stream on a bit-by-bit basis. The error sequence is generated either by a deterministic process which exactly reproduces an error sequence measured on a real communications channel or by a random process which produces an error sequence based on a stochastic model of the error sequence of a communications channel.

The significance of this approach lies in its potential application to the design of a high performance, general purpose simulator at greatly reduced cost. Use of off-the-shelf minicomputers or microprocessors and inexpensive binary random number generators promises significant savings in the cost of hardware implementation. Use of relatively simple processing algorithms promises similar savings in the cost of software development.

The proposed time-domain method treats the communications channel as a coding channel [1] which includes an encoder, a modulation channel, and a decoder. The input and output of the coding channel are digital signals. Its transmission characteristics are specified in terms of a bit-error sequence. The major processing capabilities needed to implement the time-domain method are bit manipulation, masking, counting, and random number generation. These requirements are especially well-suited for computation using standard minicomputers or microprocessors and inexpensive binary random number generators. As a result, the time-domain method promises a significant reduction in the cost of digital communications channel simulators.

The coding-channel simulator would be used to study digital communications systems and would be particularly suited for applications requiring simultaneous use of a number of communications channel simulators. For this type of application to be economically feasible, the cost of a channel simulator must be low enough so that a number of simulators can be kept on hand. Applications of this type include studies of the effect of the communications system on a larger system in which it is embedded. An example is the study of the performance of a network of computers connected via communications channels. A real-time study might involve interconnecting a number of actual computers via coding channel simulators having adjustable error characteristics. The study could provide measurements of the performance of the computer network as a function of the characteristics of one or more of the communications channels.

The alternative to the all-digital time-domain approach is the use of a frequency-domain model of the communications channel as is done in the telephone channel simulator described in Reference 2. In this approach an analog input signal is sampled and its discrete Fourier transform is computed. The Fourier transform is multiplied by a discrete Fourier transform which characterizes the channel to obtain the Fourier transform of the channel output signal. An inverse Fourier transform is computed to provide the time domain output signal of the channel.

The frequency-domain method treats the communications channel as a modulation channel [1] which includes a modulator, a transmission medium, and a demodulator. The input and output of the modulation channel are analog signals. Its transmission characteristics are specified in terms of an impulse response or a frequency response. The major processing capability needed to implement the frequency-domain method is that of computing a discrete Fourier transform in real-time. This is well beyond the capability of most general purpose computers. Due to the complex processing requirements, implementations of the frequency domain approach must employ special purpose fast-Fourier transform processors [3, 4] or special purpose signal-processing computers [5] capable of computing discrete Fourier transforms in real time. In addition, they usually include a general purpose computer to provide processing capability not available in the special purpose processor. As a result, the cost of a simulator based on the frequency domain approach is quite expensive.

A modulation-channel simulator can be used to study analog communications systems and, with the addition of an encoder and a decoder, digital communications systems. It is particularly suited to studies of the performance of particular modems or of the effect of the transmission medium on transmitted waveforms. Because of its high cost, it is poorly suited for applications requiring simultaneous use of a number of communications channel simulators.

The Time-Domain Method. The proposed time domain method simulates a digital communications channel on a bits-in, bits-out basis. That is, for an ideal channel, the bit stream observed at the output of the channel is identical to the bit stream observed at its input. The non-ideal aspects of the communications channel are observed in the form of bit errors--difference between the input and output bit streams --which can be characterized on a statistical basis. The primary function of a digital communications channel simulator is that of introducing errors into the bit stream in real time on a bit-by-bit basis in a manner which approximates the bit-error sequence of a real communications channel. The statistics of the error sequence can be determined by analytic means, by measurements of real systems, by non-real-time simulation on a digital computer, or by measurements on another media simulator. References 6 to 9 describe a number of analytic error models. References 9 to 13 present results of measurements made on data transmission facilities of the Bell System.

Figure 1 illustrates the time-domain approach proposed for introducing bit errors into a bit stream. The channel simulator receives a bit from an input bit stream, introduces an error into the bit if appropriate, and passes the bit on to the output bit stream. The error sequence introduced into the bit stream is generated by either a deterministic process which reproduces a known error sequence or a random process which produces an error sequence based on a stochastic model of the channel’s error sequence. The more important case involves the stochastic model. In that case, the decision of whether or not to introduce an error is made as follows. An n-bit random number generator computes a uniformly distributed random number A in the range 0 to 2^n-1. An error characteristic generator determines the probability P that an error occurs in a given bit and computes an integer threshold value B in the range 0 to 2^n-1 such that the probability that $A \geq B$, is exactly P. If $A \geq B$, an error is introduced into the bit before it is transmitted to the output stream. If $A < B$, the bit is transmitted without ereor. If the error sequence is based on a deterministic model, it can be handled quite simply by introducing errors directly into the bit stream.

Operation of the error characteristic generator can be illustrated by considering its use in generating three types of errors: random bit errors, burst errors, and time-varying random errors. Random errors with a probability of occurence P can be simulated by loading the error characteristic register with a fixed value $2^n(1\text{-}P)$. Since the length of the error characteristic register is fixed at n-bits, only 2^n values of P can be achieved exactly. This limited resolution can be minimized by making n large or it can be practically eliminated by periodically incrementing or decrementing the number in the error characteristic register by 1 to achieve an average value of $2^n(1\text{-}P)$. Burst errors, characterized by an error probability of 1/2, can be simulated by loading the error characteristic register with a value 2^{n-1}. More moderate bursts can be simulated by loading the error characteristic register with a number somewhat larger than 2^{n-1}. The duration of a burst can be controlled by controlling the period during which the relatively low value is in the error characteristic register. Time varying error characteristics can be simulated by varying the value in the error characteristic register as a function of time. Burst errors are one example of time varying error characteristics.

<u>The most striking feature of the proposed approach is the simplicity of its signal processing requirements</u>, which are reduced to random number generation, bit manipulation, and threshold level computation. None of the complex processing requirements found in the frequency domain approach, such as discrete Fourier transform calculation, are present in the time-domain approach.

While the approach illustrated in Fig. 1 is not restricted to a particular implementation, a strong argument can be made for using a small computer (minicomputer or microprocessor) to implement the error characteristic generator. A small computer provides an economical and flexible approach to generating the arbitrary error characteristics needed for high fidelity general-purpose simulation. The computer can perform the additional functions of comparing binary numbers, introducing errors into the bit stream, and generating pseudorandom numbers. The computer can also provide executive programs which assist an operator or user in interacting with a simulation during setup and execution. Finally, the computer can monitor the progress of a simulation and compute and save statistics and other measures of the performance of a simulated system.

Four computer-based implementations are discussed below in order to illustrate the use of computers with the time-domain method. These implementations, as well as others which envolve during the investigation, will be analyzed in the course of the research.

Implementation with One Computer   Figure 2a illustrates a straightforward implementation of the time-domain method. All functions are performed by the computer. For each bit in the bit stream, the computer generates a threshold value and a pseudo-random number, compares the two, and introduces a bit error if appropriate. Because the computer performs each operation in sequence, the bit rate which it can handle is limited by the amount of processing-required for the error characteristic generator and the random number generator.

Implementation with Peripheral Random Number Generator. Figure 2b illustrates an implementation similar to that in Fig. 2a except a peripheral device is used to generate random or pseudo-random numbers. This scheme can handle much higher data rates than that in Fig. 2a since the task of generating random numbers is removed from the computer. In general, the generation of an n-bit random number for each bit of data is a relatively time consuming task for the computer, taking from 10 to 1000 microseconds per random number depending of the computer's hardware and the algorithm used. In contrast, random numbers can be easily generated in a period of less than 1 microsecond per word by low-cost peripheral devices.

Implementation with Multiple Peripheral Random Number Generators. Figure 2c illustrates a scheme similar to that in Fig. 2b except a number of peripheral random number generators, error characteristic registers, and binary number comparators are used. Typically, the number of peripheral subsystems equals the word length of the computer, N. The output of a subsystem is a bit which indicates whether or not a bit error should be introduced into a corresponding bit in the data stream. The subsystem output bits are read in parallel by the computer in the form of an N-bit error mask. This scheme permits bit errors to be introduced on a word-by-word basis rather than a bit-by-bit basis and, in some cases, can increase the maximum data rate which can be handled by a factor equal to the word length of the computer.

Implementation with Two-Computers. Figure 2d illustrates a scheme in which two computers are used. One handles the data stream and introduces bit errors. The second serves as the error characteristic generator and computes the values which are loaded into the error characteristic registers. Peripheral random number generators, error characteristic registers, and binary number comparators are used. This scheme can handle higher data rates than that in Fig. Zc since the tasks of introducing bit errors into the bit stream, generating random numbers, and generating errors characteristics are carried out in parallel.

Performance Factors. The main factors affecting the bit rate and fidelity of a simulator based on the time-domain approach are the processing and I/O handling capabilities of the computer used, the architectural complexity of the simulator, and the complexity and fidelity of the error generating algorithm. In the simplest implementation discussed above, as the bit rate is increased, the computer eventually becomes compute-bound when it must devote all of its time to the tasks of random number generation and error characteristic generation. The maximum bit rate depends directly on the processing requirements of the error model used. In the two-computer implementation, the limiting factor is more difficult to specify. If the error model is a relatively simple one such as a Gilbert model or more generally a model which involves relatively low-frequency switching between states having fixed error rates, the limiting factors are the processing and I/O handling capability of the data-stream computer. The upper limit on the bit rate for a computer having N bits per word and N peripheral subsystems depends on the sum of the times required to input the bit- stream word and the bit-error word, to introduce the bit errors into the bit stream, and to output the bit, stream word. This limit is above one million bits per second for most modern minicomputers. If the error model requires that the values in the error characteristic registers be changed at a frequency which approaches the bit rate of the data stream, the limiting factors become the processing and I/O handling capabilities of the computer which generates the error characteristics. The upper limit depends on the time to compute and output a value for one of the error characteristic registers.

The fidelity of the time-domain simulator depends entirely on the fidelity of the error model employed. In the area of channel modeling, the frequency-domain approach appears to be better developed than the time-domain approach, possibly due to a heavier concentration of effort in the past. Most of the bit-error models available are relatively simple ones. However, new models are being developed [6] and it seems likely that improved models will become available.

Conclusions. An all-digital time-domain approach for real-time simulation of digital channels has been suggested. Four possible implementations which employ standard small computers have been presented. The approach is completely different from that used in most current channel simulators and should permit the design of a high performance simulator at greatly reduced cost. As a result the approach is particularly suited to applications requiring simultaneous use of a number of simulators. The limiting factors on the maximum bit rate which can be handled depend on the particular implementation used. The limiting factors include the complexity of the bit-error model used and the processing and I/O handling capabilities of the computers used. The limiting factor on the fidelity of the simulator is the error model employed. Here, there is a need for additional work in developing new and better error models.

References.

1. Lucky, R. W. , et al. : Principles of Data Communications, McGraw-Hill, New York, 1968.

2. Zaorski, R. W.: Digital Wireline Channel Simulator, Conference Record: IEEE International Conference on Communications, Boulder, Colorado, 1969.

3. Bergland, G. D.: Fast Fourier Transform Hardware Implementations An Overview, IEEE Trans. on Audio and Electroacoustics, June 1969.

4. Bergland, G. D.: Fast Fourier Transform Hardware Implementations A Survey, IEEE Trans. on Audio and Electroacoustics, June 1969.

5. Allen, J. : Computer Architecture for Signal Processing, Proceedings of the IEEE, April 1975.

6. Tsai, S.: Comparison Between Partitioned Markov Chain Model and Markov Gap Model for Troposcatter Channel, Proceedings of Sixth Annual Pittsburgh Conference on Modeling and Simulation, April 1975.

7. Gilbert, E. N.: Capacity of a Burst-Noise Channel, BSTJ, September 1960.

8. Elliott, E. O.: A Model of the Switched Telephone Network for Data Communications, BSTJ, January 1965.

9. McCullough, R.H.: The Binary.Regenerative Channel, BSTJ, October 1968.

10. Balkovic, M. D. , et al. : 1960-70 Connection Survey: High-Speed Voiceband Data Transmission Performance on the Switched Telecommunications Network, BSTJ, April 1971.

11. Fleming, H. C. and R. M. Hutchinson, Jr.: 1969-70 Connection Survey Low-Speed Data Transmission Performance on the Switched Telecommunications Network, BSTJ, April 1971.

12. Alexander, A.A. , et al. : Capabilities of the Telephone Network for Data Transmission, BSTJ, May 1960.

13. Townsend, R. L. and R. N. Watts: Effectiveness of Error Control in Data Communications Over the Switched Telephone Network, BSTJ, November 1964.

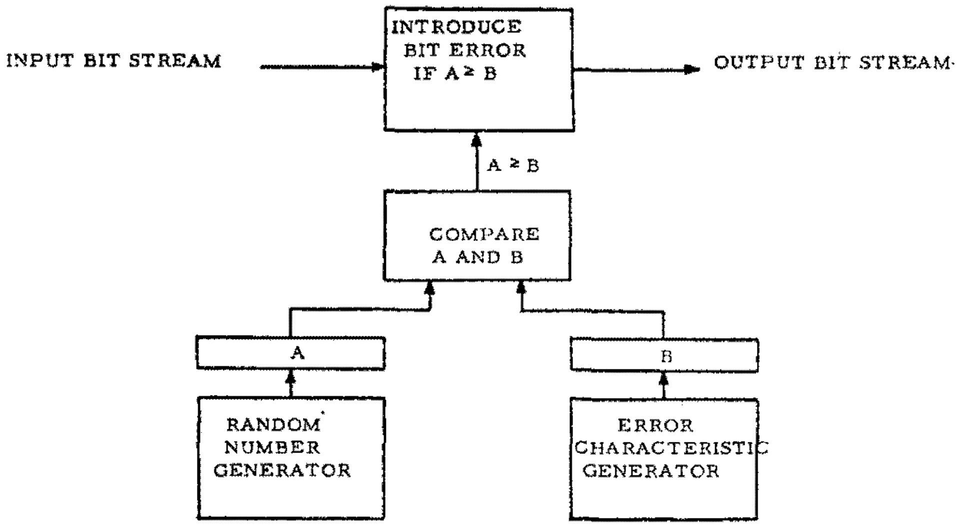

Figure 1: Basic Approach for Introducing Bit Errors into a Bit Stream

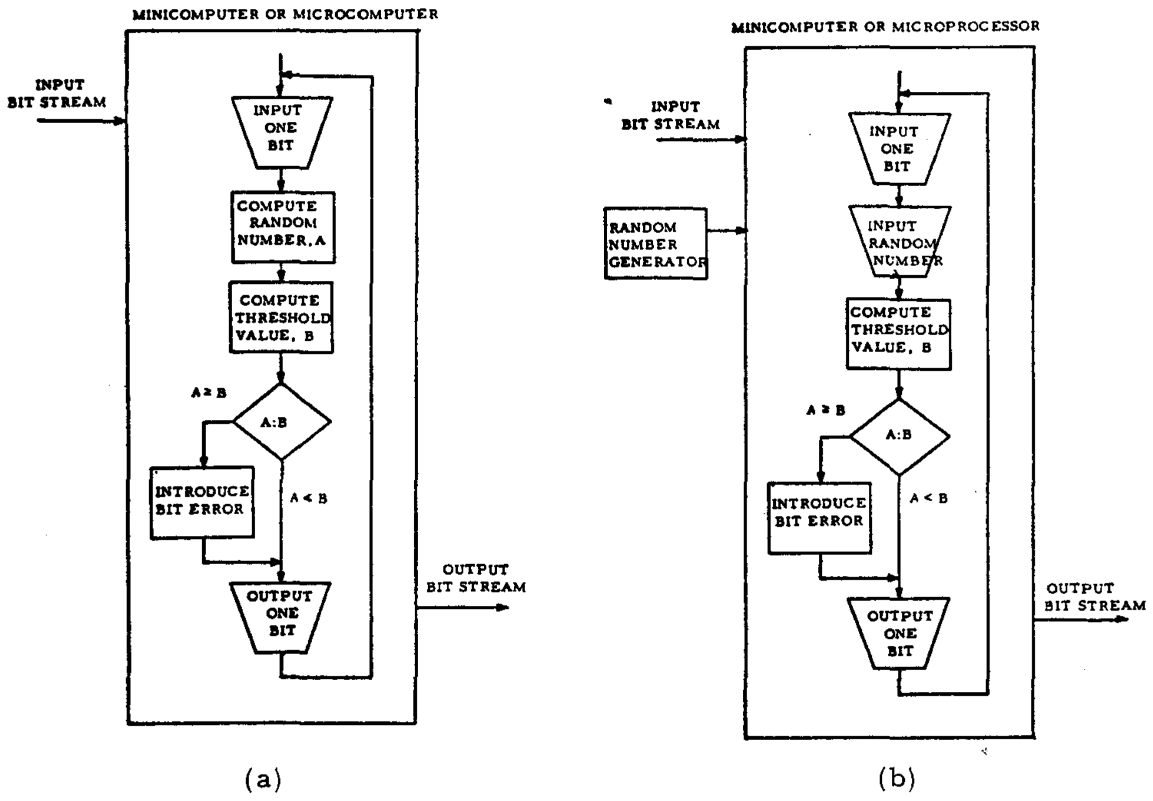

Figure 2: (a) Implementation with One Computer, (b) Implementation with Peripheral Random Number Generator

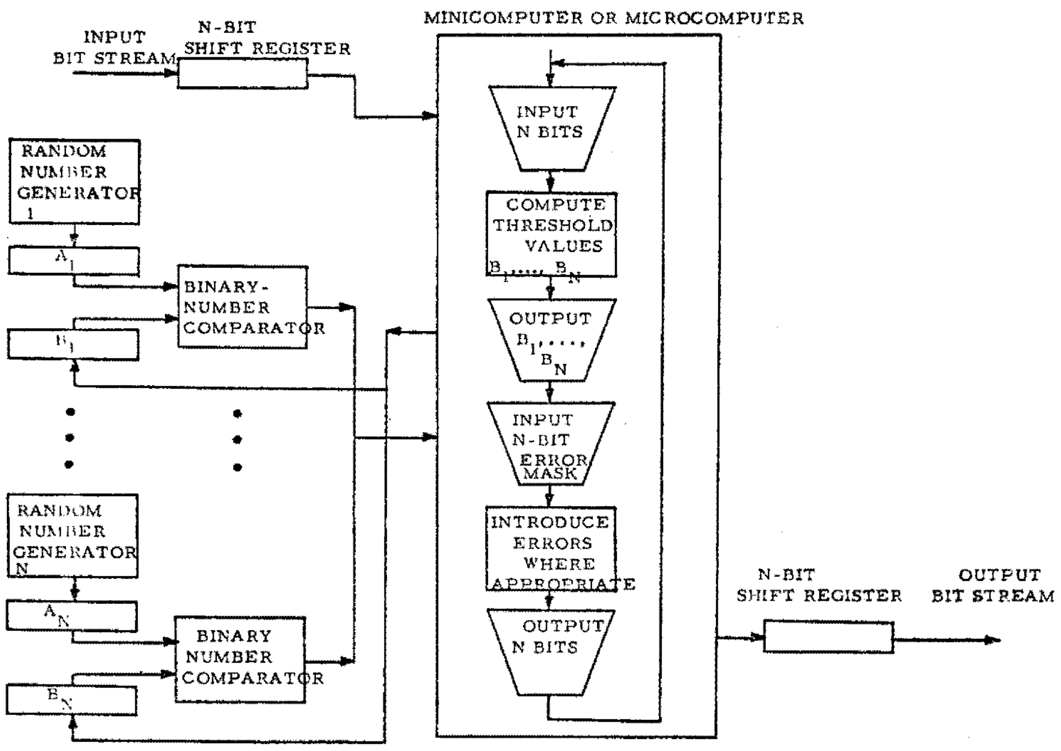

Figure 2(c): Implementation with Multiple Peripheral Random Number Generators

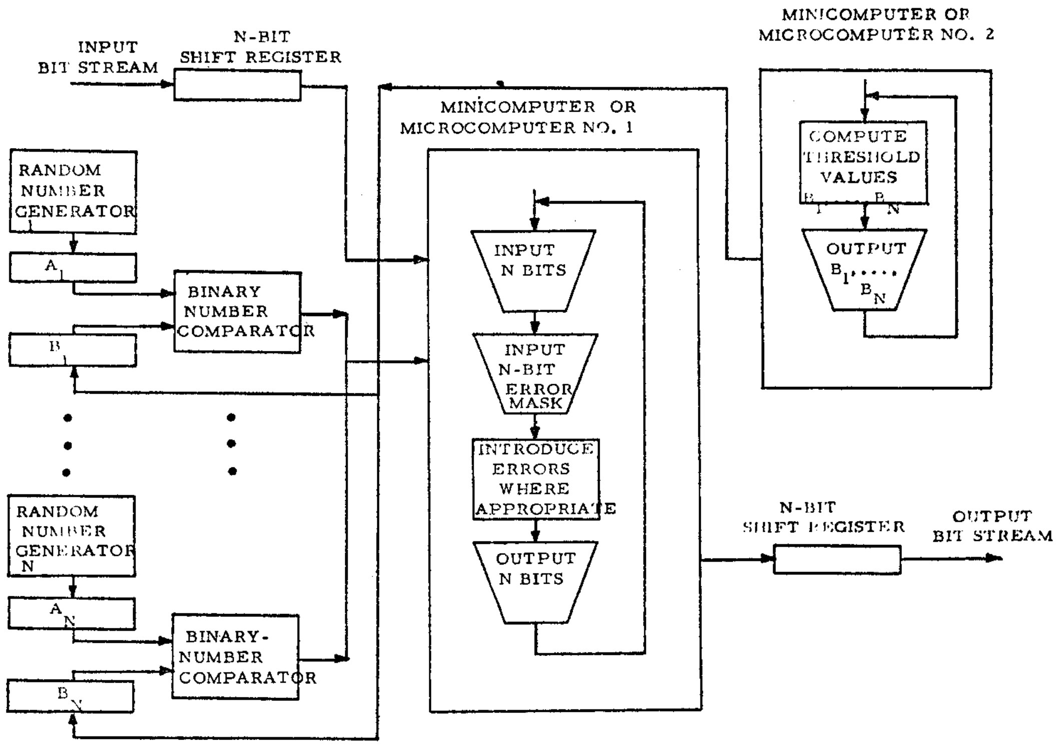

Figure 2(d): Implementation with Two Computers