USING SIMULATION TO ANALYZE PULSE STUFFING NETWORK JITTER

J. A. Gracia and R. M. Huhn
Harris Corporation
Electronic Systems Division
Melbourne, Florida

ABSTRACT

It's all about timing. In the digital communications world it is necessary to transport information over distances great and small. In order to accomplish this, it becomes imperative that timing information be made available at the receiver so that the digital information can be recovered as error free as possible. Pulse stuffing synchronization is an established technique. Hardware has been built, deployed and is operating properly in the field. The lack of a precise understanding of the phase jitter characteristics of a pulse stuffing network has created problems when interfacing special digital equipment of this type in a digital communications system. In an effort to overcome this interface problem, it became necessary to develop a better understanding of the phase jitter characteristics as a function of the parameters associated with various pulse stuffing systems now available.

This paper discusses a study which undertook such a task and in particular, addresses the role played by a simulation which was key in this study. The problem itself, much too difficult for mathematical analysis, was ideally suited for the combined simulation approach. The Multiplexer Jitter (MUXJIT) Simulator development was sponsored by the Defense Communication Engineering Center (DCEC) for the Defense Communication Agency (DCA) as part of contract DCA-100-76-C-0072 to conduct a systems analysis (and construct a supporting hardware simulator) of phase jitter characteristics in pulse stuffing time division multiplexer networks(4). The study effort was further enhanced by the construction of the hardware simulator which provided: a) verification of the preliminary mathematical and software simulation results, b) further depth by forcing the study to look at results from a hardware viewpoint, and 3) insight into hardware implementation limitations.

This effort provides a benchmark for the role of a combined software simulation and a hardware simulation effort as well as a view of the software simulator role in bridging the gap between mathematical analysis and hardware breadboarding.

INTRODUCTION

The system to be simulated, a digital telephone network, handles the continuous transfer of digital data with varying discrete corrections (pulse stuffing) which serves to synchronize the various channels so they can be combined into a single trunk line. It is cost effective to combine (multiplex) the various digital streams into a main trunk line to avoid wasting channel resources. A method of accomplishing this multiplexing is to sample from each of the tributary channels at a rate equal to the tributary data rate while the combined rate is equal to the sum total of all tributary channel rates. If all the tributary channel rates are controlled by a single clock (synchronous) and the same clock is used to control the sampling rate, the problem of combining the channels into a single trunk channel is simplified. However, due to the complexity of the digital networks and the geographic distances between the tributary sources, the synchronous approach becomes an extremely complex scheme. It is therefore often necessary to have each tributary stream run under its own clock (plesiochronous). However, in this plesiochronous scheme, the difference in clock rate, no matter how miniscule will eventually result in the trunk channel gaining or losing data bits. A conceptually simple scheme, pulse stuffing synchronization, becomes an attractive solution to this problem. Pulse stuffing basically samples the channel at a rate slightly higher than the channel's upper bound and uses filler (stuff) bits when the trunk channel gains a bit over the tributary input channel. However, when the trunk channel must be separated into its distributary channel, it becomes

necessary to accurately recreate the original channel rate so that the channel receiver can accept data as error free as possible. Hardware schemes presently used to accomplish the re-clocking of channel data have some residual jitter from that introduced when removing the stuffing bit from the data stream.

The objective of this study effort was to characterize pulse stuffing and waiting time jitter and to determine the additional effect, if any, on jitter created by cascading pulse stuffing multiplexers. Furthermore, additional motives were to apply this information to analyze and evaluate models of existing hardware to broaden the overall understanding of the pulse stuffing environment so that realistic jitter specification techniques could be derived. The approach taken to accomplish such an analysis was to develop three tools: a mathematical model, a software simulation and a hardware simulator. The preliminary mathematical analysis provided the basis to guide the effort; software simulation provided the depth to analyze the system performance; and the hardware simulator rendered confirmation of the simulation results and added the hardware depth that is often omitted from pure analysis. In addition, the interplay between the three tools provided growth to each tool, and the combination contributed to a more in depth understanding of the problem.

SYSTEM DEFINITION

A literature search (1,2,5,7) was first undertaken to identify the state of the art of pulse stuffing approaches and to identify existing pertinent first level and second level multiplexer parameters. For the purpose of this paper, a second level multiplexer is defined as one which receives and/or disperses sequential streams of digital data with each stream controlled by its own independent clock whose rate is plesiochronous with the other rates in the multiplexer. A first level multiplexer is consequently defined as one which receives and/or disperses data under control of one master clock. Data routed through a typically deployed system passes through a first level multiplexer, which collects voice, teletype, telemetry, source data, etc., followed by a cascaded sequence of second level multiplexers/demultiplexers which permit routing and transportation of the data through the complex network maze to its destination, and a first level demultiplexer which delivers the data to its destination. A first level multiplexer was involved in this study because it is the final recipient of the data handled by the second level multiplexers. Its function is to synchronize and track the data stream without data loss (bit slip). To clarify terminology, a multiplexer unit is defined to have the capability to handle both the multiplexing and demultiplexing functions.

The jitter effects considered in this study were those caused by using pulse stuffing techniques to attain data stream synchronization. More specifically, the jitter caused by removing the stuffed bit and using a smoothing loop which does not compensate perfectly for the jitter created in the data stream by the absence of the bit was the thesis of this investigation. Other effects such as a) jitter caused by the uncertainty of extracting clock from biphase or NRZ data without a companion timing signal and, b) variations in data rates such as doppler effect (if the link is relayed via satellite), environmental influences, etc., while pertinent to the overall jitter problem were not considered within the scope or objectives of this study. The primary objectives were to study the effects of pulse stuffing on clock timing, which drives and imparts jitter to the data stream, to characterize this jitter effect, and to contribute techniques to specify and measure pulse stuffing jitter.

It was postulated that in applications of interest the output data stream from the first level multiplexer could be encrypted data. In order to recover the information transmitted, it is imperative that bit count integrity be preserved in the data stream transmission until the data reached the decrypting device in the first level demultiplexer. Other criteria such as the effect of jitter on voice channel performance were considered beyond the scope of the study. Initial calculations indicated that a moderate amount of jitter would not seriously degrade voice quality. It was also observed that criteria to determine acceptable/unacceptable voice degradation is difficult to attain since voice has to be degraded severely before it is unintelligible. Consequently, bit count integrity was the criteria considered to determine acceptable/unacceptable jitter.

The survey identified the following categories of pulse stuffing multiplexers and their definitions:

a) Positive Pulse Stuffing (Positive Justification)

The provision for a fixed number of dedicated time slots (normally at regular

intervals) in the transport digital stream marks this category. These time slots are used to transmit either information from the tributary channels, or no information, according to the relative bit rates of an individual tributary channel and the transport digital signal. Additional inserted bits are deleted before the distributary channel is output.

b) Negative Pulse Stuffing (Negative Justification)

The controlled deletion of bits from the tributary channel digital signal at dedicated time slot locations is characteristic of this category. The bit rate of an individual tributary channel exceeds its corresponding bit rate in the transport digital signal. The information content of the deleted bit is carried in an overhead bit of the transport digital signal and a bit containing this information is reinserted before the distributary channel is output.

c) Positive-Zero-Negative Pulse Stuffing (Positive-Zero-Negative Justification)

This is a combination of the two prior techniques where the bit rate of the transport digital signal is usually equal to the mean tributary channel rate and either a positive or negative stuffing is used depending upon the variation of the digital rate of the tributary channel. A no stuff (zero) command is used if no rate adjustment is needed.

d) Positive-Negative Pulse Stuffing (Positive-Negative Justification)

This is an approach which differs from the prior one in that either a positive stuff or negative stuff command must be issued at every stuffing slot. Zero stuffing is achieved by alternating a positive and negative stuff. This approach only requires two commands - positive stuff and negative stuff.

The simulations, both software and hardware, were able to encompass the first three categories. The Positive-Negative Pulse Stuffing category would require some modification of the software model; the pulse stuffing portion of the hardware simulator would have to be entirely redesigned.

The survey identified a mixture of smoothing loop implementation - analog vs digital as well as first vs second order. Theoretically, the prime difference in the transfer functions of the smoothing loop is between a first and second order loop. Considerations of sampled-data theory indicated that for an oversampled system, such as this, the digital and analog approaches would yield similar results.

In regards to the first level multiplexer, the literature search revealed some basic deviations in the types of source data handled. However, in this study it was necessary to simplify the objectives by concentrating on the timing recovery aspects (bit synchronizer). From that point, first level multiplexers consisted of a one-shot window with a second-order phaselocked loop.

SYSTEM MODEL

The literature search disclosed that previous analysis efforts (1,2,5) had been performed using the classic spectrum analysis approach. In this approach the spectrum of the jittered waveform is derived by formulating the spectrum of the unfiltered jitter and, then, multiplying it by the spectrum of the smoothing loop. Calculations based on the theoretical jitter spectrum provide an upper bound on the rate at which the RMS amplitude of the cascaded jitter grows for limited cases. These theoretical calculations were backed up by test results derived from experiments with actual hardware.

Difficulty in using these results directly was encountered in relating the results to the various control parameters. The model was complex to begin with, varying given parameters was intricate and the analysis did not attempt to establish differences between stuffing jitter and waiting time jitter. Instead, it elected to deal with the combined jitter spectrum. The conclusion was reached at program onset that complementing the existing analyses with a time domain analysis might be a more fruitful approach to pursue. This conclusion was strengthened early in the study by the following four facts:

- It was easier with the time domain analysis to gain insight into the jitter phenomena. Such insight is indispensable in the trade-off study.
- The time domain analysis provided information on RMS jitter, peak-to-peak jitter, and jitter slew rates; the spectrum analysis only provided information on RMS jitter.
- Results predicted by both models were substantially in agreement.
- When model complexity was required, it was easily accommodated by the time-domain software simulation.

For the model, the three elements considered to contribute to the multiplexer jitter were:

MUX Jitter Simulation (continued)

- Stuffing jitter which results from the insertion or deletion of bits to or from the data transport bit stream.
- Waiting-time jitter which results from the situation that a pulse is not inserted/deleted when the phase difference between the data transport digit stream and the tributary input stream has changed by a bit but rather the stuffing operation is delayed until the next stuffable digit time slot.
- Cascading jitter which results from tandeming pulse stuffing multiplexers.

It was desirable to understand the elements that influence each of these jitter types so that specific control on each type could be exerted in the jitter specification to be generated.

The model involves three component elements - a multiplexer mechanism, a demultiplexer mechanism, and a smoothing loop mechanism. The multiplexer receives the tributary channel data into an elastic buffer. These data are input to the buffer second level multiplexer buffer under control of the tributary channel clock. The data are read out of the elastic buffer under control of the multiplexer's master clock. For each tributary channel a set of overhead bits and data buffer bits are carried on the multiplexer's output stream. The overhead bits carry information to the demultiplexer such as frame sync, stuff bit status or control, and in the case of negative stuffing the information (polarity) of the deleted bit. The data transport bits are composed of data bits and in the case of positive stuffing, filler or stuff bits as well. The average rate that data are read out of the elastic buffer is the data transport bit rate. The instantaneous rate is the master clock rate divided by the number of tributary channels.

The decision to insert (stuff) or delete a bit is normally rendered by phase comparator circuitry which accompanies the elastic buffer in the multiplexer tributary channel input. This decision (stuff request) is rendered by the phase comparator when the phase difference between the buffer read and write clocks is off by one bit or greater. For the channel in question, the actual stuffing operation cannot occur until a stuffing opportunity occurs. Each tributary channel is allowed an opportunity to stuff periodically. Prior to this actual stuffing operation, the overhead bits must carry information to the demultiplexer as to whether a stuff will occur. In addition, for negative stuffing one of these overhead bits (the sense digit) actually represents the "stuffing opportunity." For this case, once a stuff request is acknowledged, the next sense bit is "loaded" by the next sequential data digit in the channel. Hence, it transports the information to the demultiplexer where it is removed and reinserted sequentially into the channel's data stream.

For a positive-negative multiplexer a deviation to this algorithm is required. In this multiplexer a positive and negative stuff are alternated every other opportunity when no stuff request exists. When a request for either a positive or negative stuff arrives, the alternating sequence is interrupted and the specific requested stuff is accomplished. The alternating sequence is then resumed.

In the demultiplexer the data transport bits are written into another elastic buffer at the data transport digit rate. The multiplexer-demultiplexer data transport communications are done synchronously so the master clock of the demultiplexer is synchronized to the multiplexer rate. When a positive stuff occurs, the stuff digit is deleted by inhibiting the write clock for that bit period. When a negative stuff occurs, accompanying circuitry obtains the data bit information from the sense digit and properly inserts it into the elastic buffer in between the other data digit writes.

The bits being written into this second elastic buffer are mesochronous with the original signal since all bits are being delivered and no bit accumulation build-up is occurring in the system. (Signals are mesochronous if their corresponding significant instants occur at the same average rate. Usually, the phase relationship between corresponding significant instants varies between specified limits.) The data being written into the buffer is at an instantaneous rate which is equal to the data transport digit rate. This rate is different from the original data rate of the tributary channel. The rates are made mesochronous by the dramatic discontinuities in the data stream (missing clock bits and double clock bits). Since this jittered clock rate is not acceptable for most devices (first level multiplexers, decryptors, etc.), a phaselocked "smoothing" loop (3) is used to read the data from the elastic buffer. Since data channels are plesiochronous among themselves, each channel is required to have its own separate smoothing loop. The mission of the smoothing loop is not to track the signal, as is customary for

phaselocked loops, but rather to attempt to duplicate a fairly stable source (the input signal). Consequently, the smoothing loop is picked to have a "sluggish" response, and, hence, filter the jitter induced by the discontinuities in the data stream.

The smoothing loop is driven by a signal derived from a phase detector which takes the difference from the elastic buffer write-clock (the jittered data transport digit rate) and the read-clock (the output of the loop itself). This error signal provides the stimulus necessary to drive and to hold the loop's bit rate at the average of the original input signal, thereby making both of these signals mesochronous. The literature survey identified a mixture of first order and second order loops (both analog and digital) used in this application.

The model for the first level multiplexer is a second order loop with a window postulated for the error signal of the phaselocked loop. The width of the window is directly proportional to the one-shot's time.

SIMULATION MODEL

The requirement for the software simulation was generated from a need to extend the analysis effort. The simulation was needed as a tool to:

- Separate out and analyze the various jitter components
- Measure the effect of various smoothing loop parameters on the combined jitter components
- Interplay a given number of mux-demux smoothing loop nodes in tandem and measure the effect of cascading jitter
- Simulate specific model configurations representative of the Bell M12, VICOM T1-4000, AN/GSC-24, and HN-74 Receive Unit

Definition of the simulator model was concurrent with that of the analysis model. The prime task is the definition of the four generic parameters of the simulator: input, output, variables of simulation, and implementation mechanism. The results of these tasks are summarized in Figure 1. A basic criterion placed on the

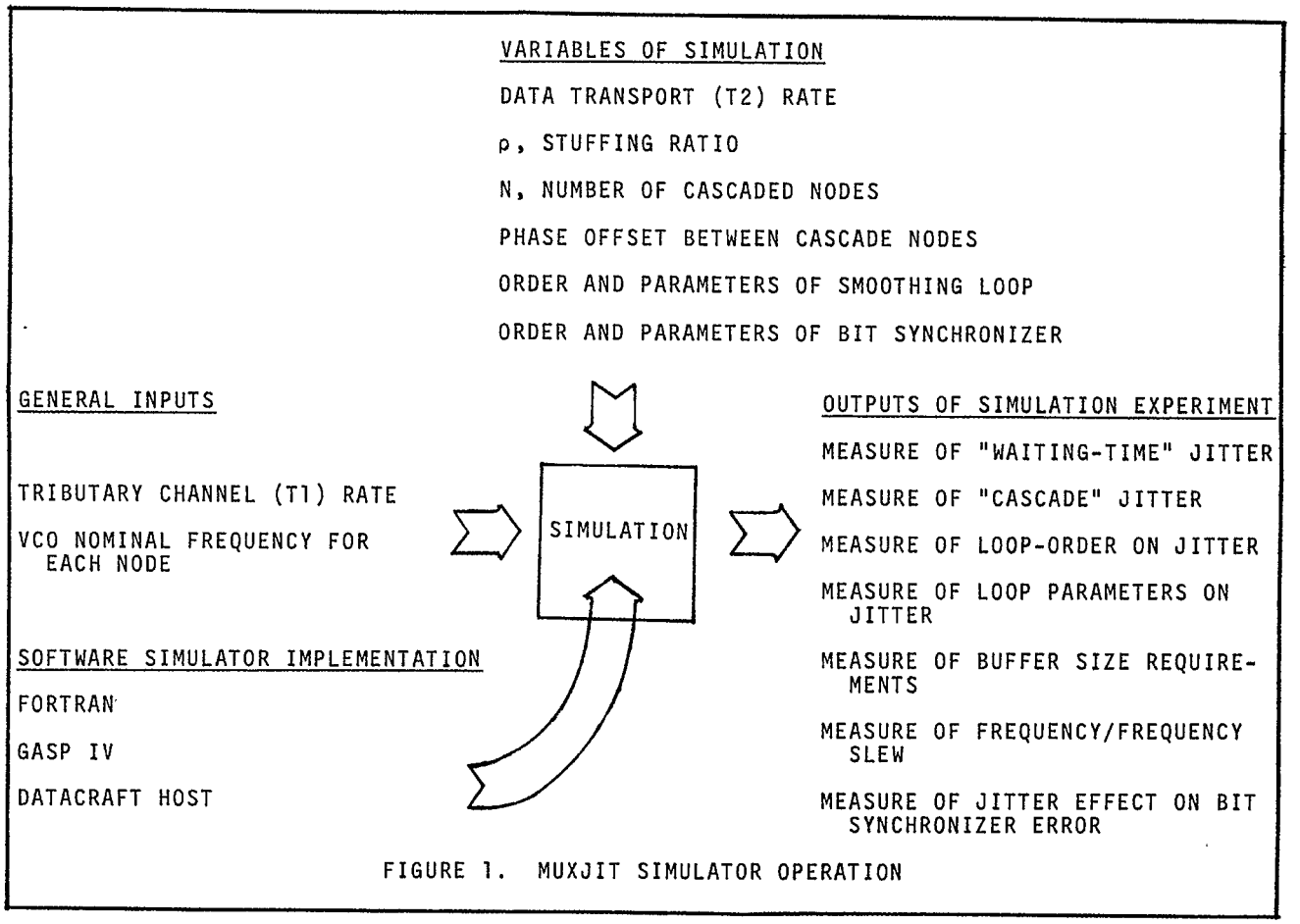


FIGURE 1. MUXJIT SIMULATOR OPERATION

software simulation was the capability to take the jittered output of a node and make that the input to another node to permit the simulation of cascaded nodes.

Figure 2 is a pictorial representation of the system model showing two nodes in cascade followed by a bit synchronizer. Each node has four frequencies of interest. These are:

a) The input frequency (f_1) which is a T1 channel tributary source for the first node.

b) The transportation frequency (f_2) for synchronous transmission between nodes called the T2 rate frequency.

c) The nominal frequency of the VCO (f_3) used in the smoothing loop for that node.

d) The output frequency which is the input frequency to the following node.

The parametric studies of the jitter cascade effect required that the simulation consist of first or second order phaselock loops at each node followed by a second phaselock loop representing the bit synchronizer.

The operation of the elastic buffer in conjunction with overhead bits to indicate a stuffing condition is modeled as a phase detector which compares the input frequency (f_1) with the node data transport frequency (f_2) and makes discrete phase corrections to the f_2 input. This corrected f_2 clock rate is an input to a phaselocked loop which models the smoothing that occurs in the system for reading data out of the elastic buffer. The phase error sampler is driven by the T2 rate. The frequency out of the first node is a jittered T1 rate which is an input to node 2. For the general case, the data stream continues through n cascaded nodes and terminates with a model of a bit synchronizer. The bit synchronizer is another phaselock loop, the difference is that the parameter of interest for the bit synchronizer is the phase error (the output of the phase lock loop's summing node or the input to the sampler) to determine total system error in detecting the proper bit in the data stream.

SIMULATION IMPLEMENTATION

The driving force in selection of the proper simulation implementation tool should be an examination of the system characteristics to be simulated. This simulation provides a good illustration of the benefits to be gained by examination of the system to be simulated.

The natural simulation model implementation for this system is a discrete simulation because the data streams are digital in nature. The smoothing loop could be implemented either as a digital or analog device. The analog device could be represented digitally; consequently, the dynamics of the phaselock loops are easily implemented as difference equations which are sampled at the T2 rate at each node. It is very tempting to make the simulation model have a one to one correspondence to the system model. For this application, it was realized that the phaselock loops were highly oversampled; i.e., the bandwidth of the PLL is very low compared to the sample rate. A simple one node model was implemented to explore the most efficient representation of the phaselock loops in terms of computer run time and simulation parameter. The model was implemented using the following three techniques: 1) difference equations sampled at the T2 rate, 2) differential equations sampled at the T2 rate with variable step size integration between samples, and 3) differential equations representing a continuous model of the phaselock loop with variable step size integration. Technique 1 provided the baseline for comparison and Technique 2 proved that the differential equations represented the same system model. Technique 3 provided an identical response to Technique 1 and ran at least 10 times faster on the computer. The variable step size integration scheme with error controls permitted relatively large time increments in the simulation after settling out from the occasional system transients created by each discrete phase correction activity.

These results provided an incentive to model the system as a combined simulation with:

- A discrete periodic phase correction based on the pulse stuffing activity.
- A continuous system model for the system dynamics using differential equations.

Accuracy in the simulation proved to be another very important implementation consideration. When performing a simulation that has error control built into the integration algorithm, it is tempting to ignore other factors that affect accuracy. A concern in this simulation was the fact that the data rates were in the MHz region (typically 1.55×10^6 Hz).

FIGURE 2. SYSTEM MODEL

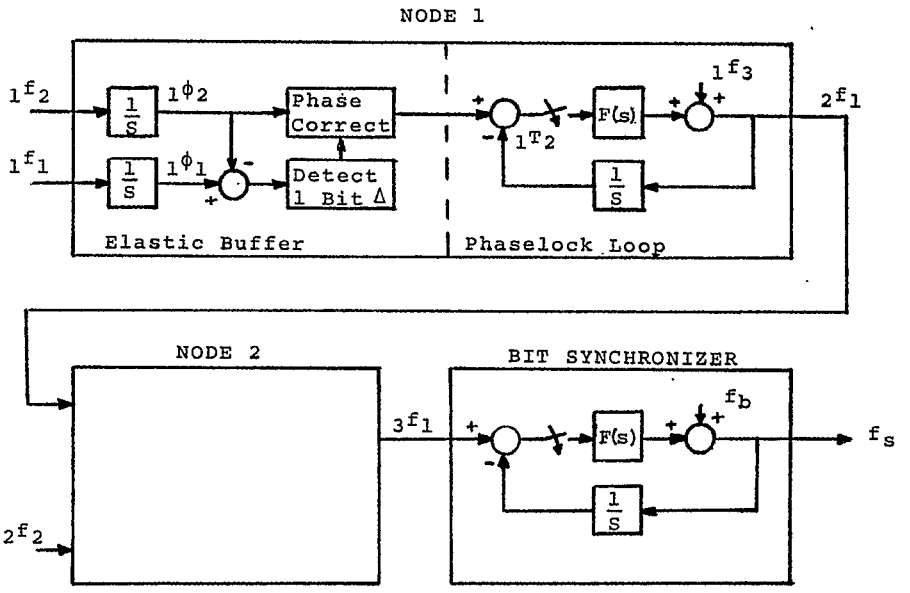
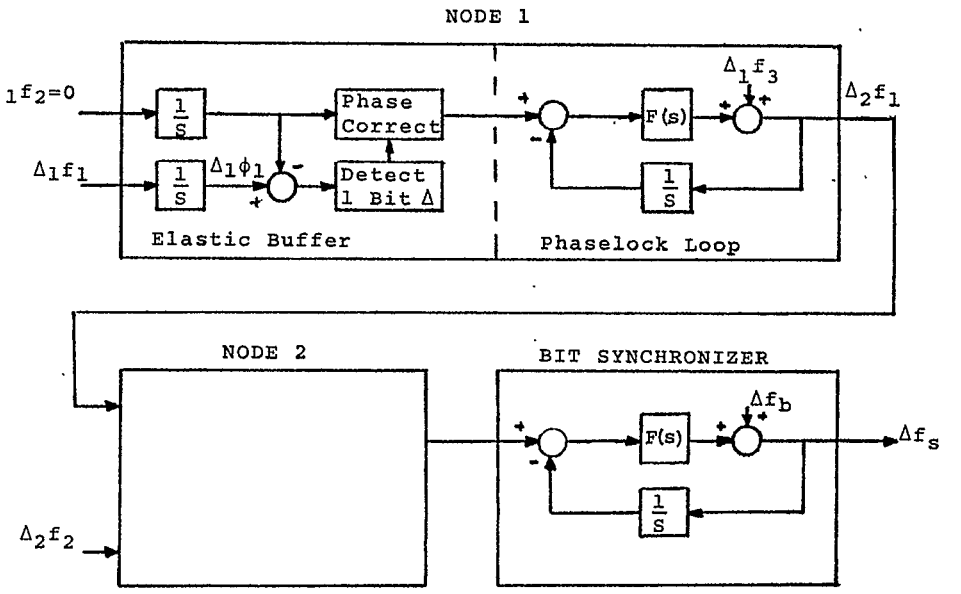


FIGURE 3. SYSTEM PERTURBATION MODEL



Since study of the steady state performance was the desired simulation output, the nonlinear performance of the phase comparator during the acquisition phase could be ignored. This permitted the system model to be implemented as a linear perturbational model as seen in Figure 3. Frequency range values in the revised simulation model were reduced to numbers on the order of 10^2 rather than 10^6.

Another factor considered in this simulation implementation was the relatively long time for the simulation to reach steady state before the desired data could be collected. Steady state as used implies hold-in performance with the repeated transients due to the pulse stuffing as opposed to signal acquisition. The initial states of the system were precalculated based on the known system perturbations (inputs to the simulation) and used to initialize the system essentially in steady state conditions.

The justification of the choice of the proper language or simulation package for a given application is a highly controversial issue primarily because of user prejudice and familiarity. Gasp IV (6), a FORTRAN based simulation language, was chosen as the tool for implementation of the system model. The features of GASP for this application were:

- Combined (discrete and continuous) simulation capability.
- Flexibility permits variable model structure and choice of outputs.
- Exact control over the sequence of events in the simulation.
- FORTRAN compatibility.

The Modeling and Simulation Group at Harris Electronic Systems Division has used GASP for complex system simulations over a period of four years with highly successful results. GASP has demonstrated reduced time (and cost) for simulation model development and verification of flexible simulations that require a large volume of computer runs on a production basis as was the case here.

Rather than modify the system equations for each system to be analyzed, a generalized simulation was written to analyze systems consisting of from one to ten nodes with first or second order phaselock loops followed by a second order bit synchronizer. The dynamic equations are table driven to permit generalized routines for first and second order nodes that are accessed based on the number of nodes being simulated for any one simulation run.

The input to MUXJIT is user oriented in that all of the variables of simulation are input in a form that is consistent with the user's thought processes. Inputs such as bandwidth, damping, etc., are converted by the program into the correct gain factors in the differential equations.

The output from the program is very flexible in that the user can selectively request up to 22 simulation outputs consisting of: plots, observation statistics, time averaged statistics, and histograms under control of GASP input data cards. Data collection starts and stops at frames specified by input data cards. The following data is collected for the nth node in the simulation:

- Input to the node
- Unsmoothed output prior to the smoothing loop
- The smoothed output
- Frequency and frequency slew variations
- Tracking error in the bit synchronizer

The simulation consists of a user written main program and five subroutines. Four of these subroutines EVNTS, INTLC, SCOND, and STATE are required to interface this combined simulation to the GASP IV Library. The fifth subroutine PHASE was needed to support the EVNTS, INTLC, and STATE subroutines. The user written routines provide the following functions.

a) Subroutine INTLC

Reads input data for variables of simulation and converts to simulation required forms, documents the system configuration, initializes the system in steady state and creates the first system events.

b) Subroutine EVNTS

The discrete events are: the periodic sampling of the phase error every N bits for detection of a stuff bit, the scheduling of the correction in phase deferred by M bits according to the particular system being simulated and the periodic saving of data only at regular intervals. The state events in the simulation occur at minimum and maximum system output phase peaks (inflection points) to permit collection of statistics on minimum phase and maximum phase variations.

TABLE 1. INTERRELATIONSHIP OF THE STUDY TOOLS

APPROACH	USES/ADVANTAGES	LIMITATIONS	COMMENT
MATHEMATICAL ANALYSIS	Derived meaningful results on effect of smoothing loop on peak and RMS "stuffing" jitter, and effect on bit synchronizer.	Difficult to analyze sampled-data system with discrete-time input as required by "waiting-time" jitter and cascading effects.	Preliminary results helpful in verifying software simulation.
SOFTWARE SIMULATION	Provided basis for formulation of "waiting-time" and "cascading" jitter. Flexible test bed for parametric analysis.	Run time limitations for quantity of results.	Results helpful in determining hardware simulator experiments desired. Provide insight to jitter specifications.
HARDWARE SIMULATOR	Extended analysis with hardware constraints. No run time limitation.	Not flexible to encompass new approaches. Limited in number of parameters which could be measured.	Provide credibility to software simulation results.

c) Subroutine STATE

Contains the generalized state equations for the first and second order nodes and the state equations for the bit synchronizer in the last node. These equations are controlled by pointers to the proper locations in the state arrays.

d) Subroutine SCOND

Contains the pointers to the system phase states to be monitored and function KROSS to detect the phase minimum and maximum peaks. The detection of these peaks automatically triggers a state event.

e) Subroutine PHASE

This is a support routine which provides the current values of phase for the nth node and bit synchronizer to the simulation for initialization and data collection.

ASSESSMENT OF PERFORMANCE

The study, in essence, was successful. Results provided an understanding of the phenomena of pulse stuff jitter and ways to quantize its effects. In addition, the impact of pulse stuffing jitter on first order multiplexers was parameterized. The effort was not scoped to extend the parameterization effort to tradeoffs between jitter and noise on the receiver bit error rate. These tradeoffs are needed to relate bit error rate specification requirements to bit synchronizer bandwidth requirements and smoothing loop bandwidth requirements. However, this extension appears feasible using MUXJIT.

The study provided an opportunity to observe three basic tools of engineering analysis and their interaction - mathematical analysis, combined software simulation, and hardware simulation. The software simulation was the heart of the effort, in fact both the mathematical analysis and the hardware simulator basically served support roles in providing depth to the software simulation. Answers to key questions...Which key parameters should be monitored?, How do you measure phase jitter?...fell out of the software simulation effort. A key contribution on the method of measurement and monitoring phase jitter via an oscilloscope display using a specially designed phase detector became obvious after observing the simulation output plot. This technique became an expedient way of obtaining hardware measurements - both for the simulator as well as fielded equipment. A comparison of the role of the three tools is summarized in Table 1 above.

In addition, this study provided a testimonial for the economics of using computer simulation in the engineering field. The use of off-the-shelf simulation packages

which are accepted industry wide, such as GASP IV, provides a means for doing sophisticated simulation rapidly and economically. It is apparent to the study team that computer simulation for such sophisticated applications have emerged out of the dungeons of expensive large manpower efforts to become an economical bridge between mathematical analysis and hardware breadboarding.

BIBLIOGRAPHY

1. Chow, P. E. K., "Jitter Due to Pulse Stuffing Synchronization," IEEE Trans. on Communications, July 1973, pp. 854-859.

2. Duttweiler, D. L., "Waiting Time Jitter," Bell Syst. Tech. J., Vol. 51, Jan 1972.

3. Gardner, F. M., Phaselock Techniques, John Wiley & Sons, Inc. New York, 1966.

4. Gracia, J. A. and T. G. Zogakis, "Analysis of Jitter Characteristics in a Pulse Stuffing TDM Network," (DCA -100-76-C-0072), DCA-TR-XXX, July 1977.

5. Matsuura, Y., S. Kozuka, and K. Yuki, "Jitter Characteristics of Pulse Stuffing Synchronization",: in Conf. Rec., 1968, IEEE Int. Conf. Communications, pp. 259-264.

6. Pritsker, A.A.B., The GASP IV Simulation Language, Wiley-Interscience, New York, 1974.

7. The International Telegraph and Telephone Consultative Committee (CCITT). Blue Book Volume III, Recommendations and Questions-Special Study Group D, 1972