it was generated, and unless already known, the width of the algorithm corridor. Several different approaches have been considered for implementation in this system. Usually, the use of longer word lengths can simplify processing but reduces the rate at which samples can be emptied from the queue and transmitted over a data circuit of fixed capacity. Shorter, more complex words will increase processing but increase the sample transmission rate.

An additional complication is that most data are presented to the flight controllers on read-over tabular displays. Changes in parameters which occur at a rate faster than once per second would probably not be absorbed by the display viewer. Therefore, the sample should not be sent unless a time plot is used for the reconstruction process and time tag lengths would decrease accordingly.

### Queues and Their Control

Significant samples occur randomly whenever a sensor data point falls outside the corridor projection of the zero-order compression algorithm. These samples are stored in a queue which is emptied at a fixed rate according to the available capacity of the NASCOM circuits. Because of the real-time nature of the system and the high-compression ratios, the conflicting requirements for short queues and a low probability of queue overflow must be satisfied.

Assume that each significant sample is described by a 30-bit word and that 4800 bps of HSD transmission capacity is available. Of this capacity, 1393 bps would be used for message overhead, error control, and uncompressed data. Therefore, 3407 bps, or 113 significant samples, could be emptied from the queue every second. To avoid queue overflows, the average activity of the compressed data must be held below this level. One way of doing this is to use a sufficiently wide corridor tolerance for the analog data. Figure 3 shows an estimate of how the average total activity of all the compressed data would vary for different tolerances applied to the analog data. The activity curve can be compared with the "clip level" of the 113 s/s read-out to arrive at an estimate for the expected range of tolerances for the analog data which, for this case, would be between 1% and 3%. This curve also shows the sensitivity of these tolerance ranges to the queue read-out rate.

However, short bursts of highly active data could force the analog-data tolerances to the upper range of acceptable values without preventing the danger of overflow. In this case, a priority list would be used to prevent some measurements from being processed by the compression program. Only significant samples from critical data would be allowed to enter the queue. In some instances, processing would stop completely until the queue length has decreased to a safe level.

Another consideration in the design of the queue and its control techniques is the burst of significant samples which occurs when the compression algorithm for every measurement is initialized. Packed "initiation blocks" are being most favorably considered to satisfy this function.

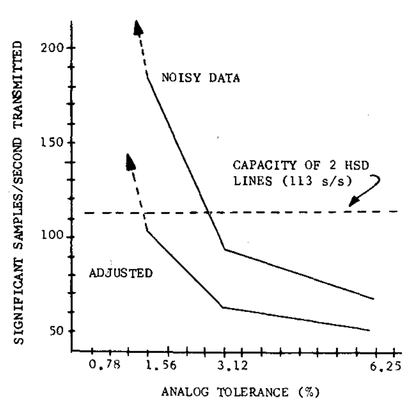


**Fig. 3 Total data activity vs analog tolerance.**

### Error Control Requirements and Techniques

Because compression ratios are so high, it is possible that only one significant sample from a measurement may be generated during an entire pass over a remote site rather than at the present maximum interval of once every ten seconds. An error-control system must be used to guarantee the error-free arrival of each significant sample.

The selected system was a type of hybrid forward-acting and retransmission system. Most communications circuits are characterized by random errors of a few bits and bursts of very high error rates. The forward acting capability of the system will correct at the reception point the random errors. In order to minimize the number of error-control parity bits sent over the circuit, bursts would be handled by an error-detection code which initiates the transmission of a repeat request to the transmission point. Retransmissions due to random errors would be prevented by the forward-acting code but a low probability of accepting bad data would be attained with the use of the error detection code and the retransmission technique. Thus, a high level of transmission reliability can be attained with a low-overhead penalty.

Various schemes are being considered, a Binary Detector Corrector based upon a N-bit Hamming encoding scheme shows much promise being faster than others evaluated. The Zierler-Gorenstein symbol code, Reed-Solomon error detection code as well as BCH error-detection devices have been considered.

### Use of Compressed Data by the Flight Controllers

Until a complete system is available for operational simulation, it is difficult to state what reactions will be generated by the flight controllers when presented with compressed data. Depending upon the performance of the system, the presence of varying queue delays, extreme variation in update arrival rate from many per second to once every several minutes, and varying tolerances may or may not be acceptable. If all these characteristics are acceptable, confidence in the system and its method of data processing and presentation will follow as the operational time increases.

### Reference

[1] Stine, L. L. and Barrows, J. T., "The Application of Data Compaction to Apollo Flight Control Telemetry Data," MTR-498, Oct. 1967, The MITRE Corp., Bedford, Mass.

# An Error-Detecting Test Using Generated Number Sequences

Richard I. Hess*
*Logicon Inc., San Pedro, Calif.*

### Introduction

BEFORE launching a space vehicle or missile it is necessary to somehow ascertain that the onboard program will perform its functions properly. Of particular importance in this regard is the verification of the targeting constants needed for guidance and other flight computations. These constants, typically a few hundred in number, must be computed before liftoff; this is necessary because generation is fairly time-consuming.

Received September 2, 1969; presented as Paper 69-946 at the AIAA Aerospace Computer Systems Conference, Los Angeles, Calif., September 8–10, 1969; revision received February 5, 1970.

* Member of the Technical Staff; also Part-Time Lecturer in Mathematics, California State College at Dominguez Hills.

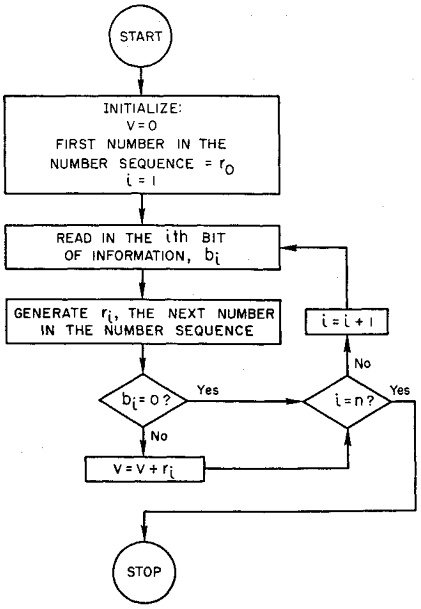


**Fig. 1 Generation of the verification parameter using a number sequence.**

It is desirable to make a detailed check of each one of these constants, but if launch must proceed without enough prior notice to allow such a detailed check to be made, the verification may have to consist solely of a series of checks performed within the onboard computer. These internal checks might range from something as exhaustive (and time-consuming) as comparing each constant with a corresponding constant generated by another computer to something as simple as storing and performing a single parity check. A procedure lying between these two extremes consists of performing a checksum test, that is, loading into the onboard computer an independently computed sum of all targeting constants and comparing it with the sum of these constants as computed onboard. The efficiency of the checksum technique will be compared with an error-detecting test using generated number sequences.

### Checksum Test

Let us consider a grossly simplified error model in order to estimate the reliability of verification with a checksum test. Suppose the targeting constants consist of $w$ words each having $b$ bits. Further suppose (and this is a simplification) that each bit has an a priori probability $p$ of being incorrect independent of the other bits. The total number of bits is $n = bw$. The probability that $k$ of these bits will be incorrect is given by the binomial probability distribution

$$P_k = \binom{n}{k} p^k (1 - p)^{n-k} \tag{1}$$

where the binomial coefficients are

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \tag{2}$$

We expect that typically the onboard computer is fairly reliable so that $P_0 > 0.99$ and $P_1 < 0.01$. What we now want to determine is the probability that an incorrect set of constants will pass the checksum test. Under the conditions of this model, by far the most likely way for this to happen occurs when 1) there are errors in exactly two bits, 2) they occur in different words at the same bit position, 3) one of the errors is a 0 where a 1 should be and the other is a 1 where a 0 should be. The over-all probability of such an occurrence is

$$P = [P_2]\cdot\left[\left(\frac{n-b}{n-1}\right)\frac{1}{b}\right]\cdot\left[\frac{1}{2}\right] = \frac{P_1^2}{4bP_0}\left(\frac{n-b}{n}\right) \approx \frac{P_1^2}{4b} \tag{3}$$

where brackets enclose the individual probabilities of each of the three conditions just identified.

If we consider a typical onboard computer with $b = 24$ and assume that $P_1 = .01$, we find that $P \approx 10^{-6}$; thus the probability that a bad set of constants will pass the checksum test seems remote. Unfortunately, this result is based on the unsupportable assumption that single-bit errors occur independently. Certain plausible error modes, perhaps occurring more often than once in $10^6$ missions, lead to a situation in which an incorrect set of constants will pass the checksum test. One example occurs when two constants are computed internally according to the formulas $C_i = a + x$, $C_j = b - x$ and the internal constant $x$ is in error. Another occurs when two constants become interchanged. What is needed is some way of reducing the probability that the effects of errors will cancel each other in the verification test. The error-detecting scheme proposed and analyzed below fulfills this need by using a generated sequence of numbers.

### Proposed Error-Detecting Scheme

The scheme shown in the flowchart of Fig. 1 may be described briefly as follows:

1) Take the entire set of $w$ constants expressed in words $b$ bits long and generate a sequence of numbers, one for each of the $n$ bits.

2) Take the sum of those numbers corresponding to bits of value 1 and do not use those corresponding to bits of value 0.

3) This sum of numbers is the verification parameter. It is entered on the flight tape and later compared to one similarly formed in the onboard computer. It is desirable, for the sequential numbers, that a) no two are the same, b) none of the numbers used is 0, c) the numbers follow some predictable sequence so that the same number is used for corresponding bits in the flight tape preparation and the onboard computer. All three of these conditions can easily be satisfied by the number sequence $r_i$ defined by the congruence

$$r_{i+1} \equiv ar_i + c(\text{mod } 2^q) \tag{4}$$

if the constants $a$, $c$, and $q$ are appropriately chosen. The effect of Eq. (4) is to generate the next number in the sequence by multiplying the present number by a constant, adding another constant, and keeping the right-hand $q$ bits. Such a generated sequence of numbers, each $q$ bits long, is sometimes called pseudorandom since the sequence has some of the properties associated with a sequence of truly random numbers.

The verification parameter may be expressed as

$$V = \sum_{i=1}^{n} r_i b_i \tag{5}$$

where $b_i$ is the value of the $i$th bit.

To compare the effectiveness of using the parameter $V$ defined in Eq. (5) with that of a checksum, it seems fair that both $V$ and the checksum should have the same number of bits as the sum of all the constants. If there are $w$ words of $b$ bits each and it is assumed that overflow beyond the $b$ bits is carried in the checksum, then the number of bits in the sum will be

$$N_{cs} \approx b + \log_2(w) - 1 \tag{6}$$

where $\log_2(w)$ is the logarithm of $w$, base 2. The number of bits in $V$ will be

$$N_v \approx q + \log_2(n) - 1 \tag{7}$$

and $q$ will be picked so that $N_v$ and $N_{cs}$ are equal

$$q \approx b + \log_2(w) - \log_2(n) \approx b - \log_2(b) \tag{8}$$

In the case for which $b = 24$, $q = 18$ is a good choice.

The effectiveness of $V$ as a verification parameter will first be analyzed by using the model given under the checksum test; then an analysis will be made using a possibly more realistic model. Results from both analyses show that the use of $V$ instead of a checksum significantly reduces the probability of undetected errors.

The conditions imposed on the sequential numbers that no two are the same and none of them is 0 imply that there must be at least three bits in error before an incorrect constant set can pass the proposed verification test using the parameter $V$. The most likely way for this to occur under the conditions of this model happens when: 1) there are errors in exactly three bits; and 2) the three $q$-bit numbers by which $V$ is in error cancel each other (one-fourth of the time these three numbers will be of the same sign and cannot cancel). The over-all probability of such an occurrence is approximately

$$P = P_3 \cdot \tfrac{3}{4} \cdot 2^{-q-1} \approx P_1{}^3 \cdot 2^{-q-4} \tag{9}$$

where the probability that two $q$-bit numbers add up to a specified third $q$-bit number is estimated to be about $2^{-q-1}$. For $P_1 = 0.01$ and $q = 18$, $P$ comes out to be $2.4 \times 10^{-13}$—a very small probability indeed.

For a 24-bit computer the probabilities of an undetected error using the checksum and the proposed test are given in Fig. 2 as a function of $P_e$, defined as the probability of at least one error ($P_e = 1 - P_0$ for this model). It is clear that for the stated model the proposed verification test is significantly superior to a simple checksum test.

The model used thus far has the property that multiple-bit failures are very unlikely. For this reason it may overrate the effectiveness of the proposed verification test. Let us now consider a model that assumes that most of the time when any error exists, that error will have propagated into many bits of error in the targeting constants.

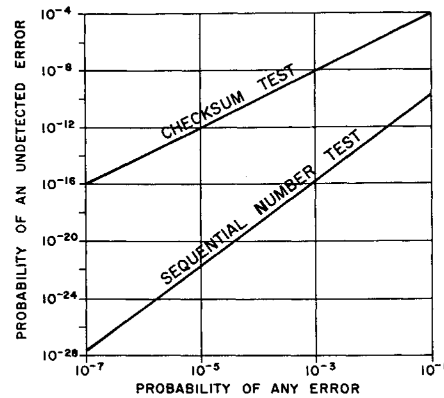


**Fig. 2 Effectiveness of checksum and sequential number tests.**

If $P_e$ is taken as the probability of any error at all, then under this model the probability that the sequential number test fails to discover an existing error is less than

$$P_{\max} = P_e \cdot 2^{-q-1} \tag{10}$$

For $P_e = 0.01$ and $q = 18$, $P_{\max}$ is about $2 \times 10^{-8}$. Even for this alternative model the proposed test is still very effective. Its chief advantage over the checksum test is that errors that defeat it are expected to occur only very rarely, whereas error modes that defeat the checksum test are fairly easy to envision.

**Table 1 Maximum periods for binary, octal, and decimal Johnson counters**

| Number of digits, $q$ | Maximum period: Binary counter | Octal counter | Decimal counter |
|---|---|---|---|
| 2 | 3 | 12 | 60 |
| 3 | 7 | 28 | 217 |
| 4 | 15 | 60 | 1560 |
| 5 | 21 | 84 | 168 |
| 6 | 63 | 252 | 196812 |
| 7 | 127 | 508 | 2480437 |
| 8 | 63 | 252 | 15624 |
| 9 | 73 | 292 | |
| 10 | 889 | 3556 | |
| 11 | 1533 | 6132 | |
| 12 | 3255 | | |
| 13 | 7905 | | |
| 14 | 11811 | | |
| 15 | 32767 | | |
| 16 | 255 | | |
| 17 | 273 | | |
| 18 | 253921 | | |
| 19 | 413385 | | |
| 20 | 761763 | | |
| 21 | 5461 | | |
| 22 | 4194303 | | |
| 23 | 2088705 | | |
| 24 | 2097151 | | |
| 25 | 10961685 | | |
| 26 | 298935 | | |
| 27 | 125829105 | | |
| 28 | 17895697 | | |
| 29 | 402653181 | | |

### Execution Time for the Verification Test

Even though the verification test requires generation of a number for each bit of information in the targeting constants, for practical applications the extra computer time required is negligible. It is still desirable, however, to reduce any time penalties by looking for a fast method of generating number sequences which do not repeat except after many terms. The congruence method described previously has the drawback that the generation of each number requires the comparatively lengthy operation of multiplication. A much faster method is given by use of a special kind of binary ring counter, sometimes referred to as a Johnson counter,[1] as described below.

Suppose the $i$th number in the sequence is given in binary form as

$$r_i = a_0 a_1 \ldots a_{q-2} a_{q-1} \tag{11}$$

where the $a$'s are either 0 or 1. The next number in the sequence is given by

$$r_{i+1} = a_{q-1} a_0 \ldots a_{q-3} a_m \tag{12}$$

where

$$a_m \equiv a_{n-2} + a_{n-1} (\text{mod } 2) \tag{13}$$

The period of the resulting sequence of numbers depends upon the starting number and $q$, the number of bits in each number. For every $q$ the maximum period, which is not simple to determine, occurs when the starting number is 1 and is given in the first column of Table 1. It is clear from the table that certain choices of $q$ should be avoided since the associated period is too small. The Johnson counter technique can also be used to generate number sequences by manipulating digits of numbers expressed in bases other than 2. Table 1 also gives the maximum periods obtained when octal and decimal number sequences are generated.

### Conclusions

A useful test has been advanced for checking the correctness of flight constants computed onboard before launching a missile or space vehicle. This test, which involves parallel computations in the onboard computer and a checking computer of a verification parameter using a generated number sequence, has been found to be orders of magnitude more effective in detecting errors than a test using the sum of the flight constants as a verification parameter.

### Reference

[1] Fitzpatrick, G. B., "Synthesis of Binary Ring Counters of Given Periods," *Journal of the Association for Computing Machinery*, Vol. 7, No. 3, July 1960, pp. 287–297.

---

# Effects of Mission Environments on the Mechanical Properties of Dacron Parachute Material

DAVID C. SPENCE*
*NASA Langley Research Center, Hampton, Va.*

DACRON is a candidate fabric for parachutes for planetary landers because of its high strength and heat stability.[1,2] It was used in Planetary Entry Parachute Program (PEPP),[3] which showed the feasibility of utilizing a parachute as a spacecraft decelerator, and a part of the fabric originally purchased for the PEPP parachutes was used in the present investigation. Although some investigators[4–7] have studied effects of temperature, vacuum, and/or sterilization treatments on candidate textile materials (Nylon, Dacron, Nomex, and Fiberglas), no one, to our knowledge, has determined the effects of an appropriate sequence of mission environment phases, without interruption, on the mechanical properties of Dacron. Moreover, some investigators have removed the fabrics from the environment under study in order to facilitate tensile testing, but others[8] have shown that in situ testing is necessary to assure that measurements of environmentally induced effects are valid. This investigation employs the in situ test philosophy throughout and covers the effects of exposure of Dacron to the sequence of prelaunch and flight environments that it might experience for a Mars-lander mission.

### Experimental Program

Two groups of Dacron samples were tensile-tested: 1) the control group, stored and tested at 1 atm, 75°F, and 45% relative humidity (RH); and 2) the group subjected tc the mission environments in the sequence given at the top of Fig. 1, whose abscissa is elapsed time in days. Although temperatures are different for the different environments, the samples were all tensile-tested at 75°F. Initially, 45 samples were exposed to chemical sterilization by ethylene oxide (ETO), using a gas mixture of 12% ETO, 88% Freon-12 at a concentration of 500 g/l, 40% RH, and 104°F for 24 hr. At the end of this treatment, 5 samples were uniaxially tensile-tested. The remaining samples underwent dry heat sterilization (DHS) at 257°F in dry nitrogen at 570 torr for 24 hr. They were then cooled to 75°F, and five more were tensile-tested. Then the chamber was evaucated to less than $10^{-6}$ torr, and another 5 samples were tested. Other sets of 5 were tested on the 3rd, 6th, 10th, 14th, and 16th days to determine the effects of the vacuum environment over a 13-day period, with the vacuum varying between $1 \times 10^{-6}$ and $5 \times 10^{-8}$ torr. On the 16th day, the pressure was brought up to 8 torr with a simulated Martian atmosphere (MA) consisting of 74.4% $CO_2$, 12.8% $N_2$, and 12.8% Ar. After a 120-sec exposure to simulate parachute deployment on entry, 5 more samples were tensile-tested. Finally, 5 samples were tested 24 hr later (17th elapsed day) to determine whether longer MA exposure had any effect.

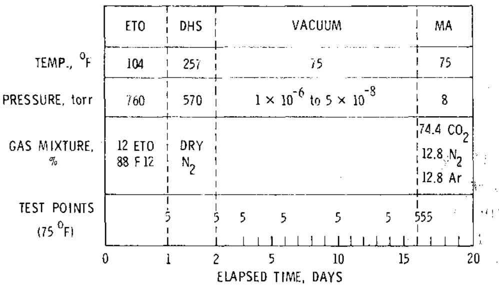


**Fig. 1 Sequence of environment phases simulated and test conditions.**

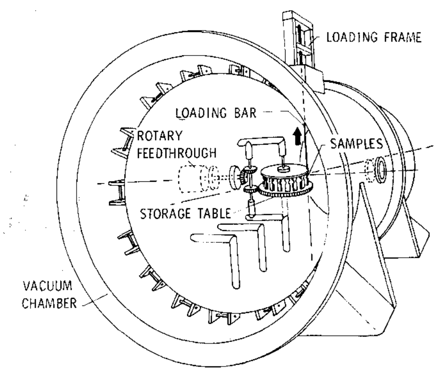


**Fig. 2 Schematic of the LRC Space Vacuum Facility with the tensile test apparatus.**

The testing took place in the $150\text{-ft}^3$ Space Vacuum Facility at Langley Research Center, using a carousel type, uniaxial tensile apparatus (Fig. 2).[9] The major items of interest are the storage table with samples, the rotary feed-through, the loading bar, and the loading frame. The samples were prepared in accordance with the ASTM raveling technique.[10] From the bulk cloth, a 1.25 × 10-in. sample was cut with the longitudinal axis parallel to the warp direction of the material and raveled in width to a constant strand count. It was then mounted in 2 × 2-in. stainless-steel tensile jaws to permit a 1 × 3-in. area to be free for environment exposure. The samples were then placed in the carousel, and at the time of tensile-testing they were continuously loaded at a crosshead speed of 2 in./min until failure occurred. The output of the load cell was externally recorded at a chart speed of 5 in./min.

### Discussion of Results

Figure 3 represents a typical stress-strain curve and shows how various mechanical properties may be calculated from it.

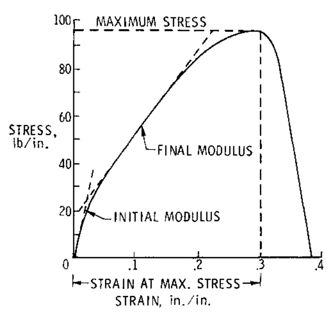


**Fig. 3 Typical stress-strain curve for ASTM type tensile test.**

Presented as Paper 69-1018 at the AIAA/ASTM/IES 4th Space Simulation Conference, Los Angeles, Calif., September 8–10, 1969; submitted September 22, 1969; revision received February 2, 1970.

* Aerospace Technologist. Member AIAA.