A New Feedback Protocol for the Ground Communications Facility

O. H. Adeyemi
Communications Systems Research Section

This report summarizes a simulation study of a feedback scheme that can reduce the Ground Communications Facility error rate by more than two orders of magnitude in real time even in the highest error mode. The after-pass retransmission time is eliminated or reduced drastically in the case of long time outages. The new scheme also provides storage for outage data, thus eliminating the search time for data to be retransmitted after-the-pass for the Clean Tape Log. No new hardware is required.

I. Introduction

Until now, our efforts to reduce the Ground Communications Facility (GCF) error rate from the present bit error rate (BER) of 10^{-4} (or block rate of about 10^{-3} to 10^{-2}) to an acceptable value of less than 10^{-6} (or block rate of less than 10^{-5}) have been hampered, first, by the discovery in 1974 that forward error correction was not effective (Ref. 1) and then, later, by objections to the long time required for necessary internal processing implicit in the two feedback retransmission schemes proposed at that time (Refs. 2 and 3). The feedback protocol proposed in Ref. 4 was designed mainly to eliminate these objections by letting our USER dictate what time he can give us for this internal processing and then finding the reduction in error rate that can be achieved as a function of this *waiting* time. It also has the important feature of providing a storage for bad data, including outage data that cannot be delivered in real time, to hold such data for retransmission during the filler block times (when the data line is clean) or after the pass in the case of long outages (a rare event, but still a problem), thus eliminating the search time for outage data. Equally important, no new hardware is required.

Real-time error reduction is not the only feature desirable in the GCF. It must be possible to log error-free data with minimum after-pass retransmission. There are, therefore, two main parameters by which to evaluate the performance of this scheme (and indeed any feedback scheme on the GCF). Given the acceptable USER waiting time, we need to know the error rate of the data delivered to him in real time and the reduction in the time for retransmission of bad data after the pass. As to the second requirement, we show that this scheme cleans up all the *normal* errors on the GCF *during* the pass, including all short outages; part of the long outage data (several thousand blocks long) may have to wait until after the pass for retransmission for the Clean Tape Record. This report is addressed principally to the first question: the real-time error rate to the USER.

In a simulation study of about 10½ hours of real-time data in the *noisiest* mode of the high-speed 4.8-kbps data line, 2000 block errors were reduced to 128 block errors to the USER at a maximum waiting time of only 3 seconds and to only 40 block errors if the maximum waiting time is increased to 6 seconds. This is a reduction in error rate from a BER of 10^{-4} to 10^{-6} at a waiting time of 3 seconds, and it is the highest USER error in all the simulations. In several cases, all the errors were corrected within the allowable waiting time. The highest errors delivered for different USER waiting times were:

Waiting time t, seconds	Error to USER
0	2000
3	128
6	40
9	31

The time $t = 0$ corresponds to no processing (no feedback retransmission).

The wideband data line is known to be better than the high-speed line, which explains the better performance of this scheme in the wideband mode at a waiting time of 3 seconds. The error rate to the USER in this case is 10^{-7}. The wideband data line performance at the noisiest mode was:

Waiting time t, seconds	Error to USER
0	2000
3	69
6	52
9	49

In both the high-speed and the wideband data line modes, 10 hours of line use provide literally thousands of filler block times at 90% (or 95%) data rate during which the 128 blocks (high speed) or the 69 blocks (wideband) for the Clean Tape Record can be retransmitted. Thus, as mentioned above, this scheme requires no after-the pass retransmission time during 99.9% of the total GCF operation comprising the normal error and short outage (up to several hundred blocks) modes.

II. The Retransmission Scheme

A schematic diagram (Fig. 1) is given to aid in understanding the feedback protocol.

The operating procedure is as follows: Each new data block transmitted is stored in the Priority Buffer (PB) until an acknowledgment signal has been received at transmitter T as to whether it has been received correctly. Thus, by the time word is received about the error status of the first block transmitted, as many new data blocks as can be transmitted within a loop time (LD blocks) will have been sent. All these LD blocks will be stored in the Priority Buffer. As soon as an acknowledgment signal indicates that a block has been received error-free, that block is dropped from the Priority Buffer. Thus, at any time not more than LD blocks are stored in this buffer, and the current feedback error status signal applies to the oldest block in it.

Now suppose an error message is received and the USER allows only t seconds for any internal processing that can be done to correct this error block and the burst that may follow. Since data must be delivered in sequence to the USER, delivery of all subsequent blocks is held up in the Receive Buffer (RB) until the error block is corrected or until t seconds are up, whichever comes first. Think of this USER waiting time in terms of the $tR_T/1200$ blocks that can be transmitted in the allowable t seconds. Better still, let us think of it in terms of the $NT = [(tR_T/1200) - LD]$ blocks that can be transmitted from T to the receiver R starting from when the error status message is received at the transmitter until the time required for a new block transmitted to reach the receiver just before the t seconds are up. The NT block times are all we can use for any possible retransmissions we must do.

When the error message reaches the transmitter T indicating the beginning of a burst:

(a) The transmitter time is set at TIME $= NT$, counting down one every block time.

(b) The incoming new data stream is diverted into the New Data (ND) Buffer.

(c) The error block is taken from the first position in the Priority Buffer, transmitted, and re-inserted in the PB but now as the newest entry. Thus, the Priority Buffer still contains the LD blocks on which error status reports have not been received.

Steps (b) and (c) are continued in the case of consecutive errors. However, if within the execution of these two steps there is a good block, then:

(i) Drop the acknowledged block off the Priority Buffer.

(ii) Transmit the oldest block of the new data stream in ND Buffer.

(iii) Insert the new block into the newest block position in the Priority Buffer.

STEPS (b) AND (c) ARE OPERATIVE ONLY IF $NT > 0$.

As soon as the retransmission time is up (when $NT = 0$):

(1) Start transmitting new data from the queue in the ND Buffer.

(2) Empty all the LD blocks in the Priority Buffer into the Old Data Buffer (ODB). These are the blocks on which error status reports have not been received and which may contain errors. When the acknowledgments, if any, are received, only the bad data are retained in the ODB. The ODB contents are transmitted within the filler block times during 99% of the time when the GCF is in the normal error-free mode. These blocks are then merged with stored error-free real-time blocks to form the Clean Tape Log.

(3) Deliver all the blocks in the Receive Buffer (though some are in error) to the USER *and* write a copy of these blocks on the Clean Tape Log. Since up to 40% of the bad blocks contain 50-bit errors or more (almost 50% in the wideband data line), not many of the error blocks delivered to USER would be useful.

In summary, we need four buffers and a storage for clean data.

(1) The Receive Buffer stores all blocks correctly received during retransmission time until all prior blocks have been successfully retransmitted or until the retransmission time is up, whichever comes first.

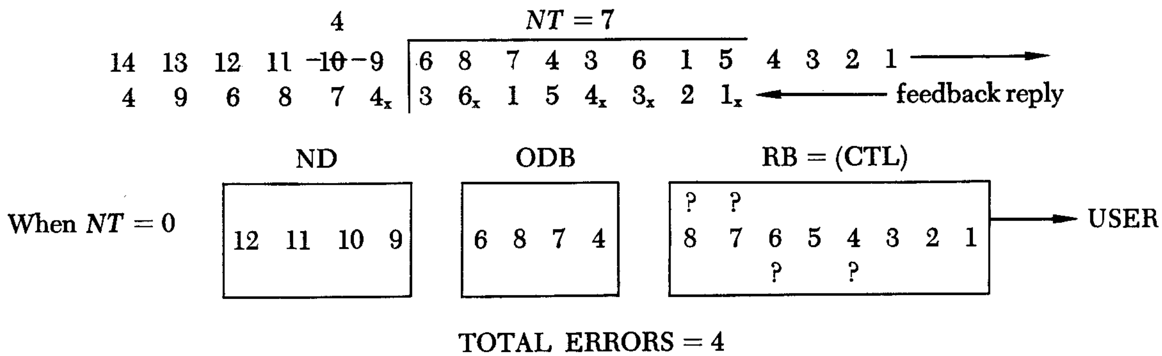
(2) The New Data Buffer stores the new data stream during the retransmissions.

(3) The Priority Buffer retains each new block until its error status report is received and each retransmitted block until the retransmission time is over, then all the contents are emptied into the ODB.

(4) The Old Data Buffer holds all the data that cannot be delivered in real time. During periods of long line outages, this buffer becomes increasingly long as undelivered data continue to pour into it from the Priority Buffer. The Priority Buffer and the Old Data Buffer can be part of the same physical device with two pointers: one to the data that can still be corrected before they are delivered to the USER, and the other to those data that must wait until filler block times or after-the-pass.

(5) The Clean Tape Log is a record of only correct data. It acquires error-free data by merging real-time good blocks with those retransmitted later from the ODB.

The following is an example of how the scheme works:



This is an example of high-speed 4.8-kbps data line performance for which the USER allows up to 3 seconds for retransmissions. The 3 seconds is a total of 12 block times. Suppose the loop time is equivalent to 5 block times; then we have 7 block times for all the retransmissions we may need to do. When block 5 is being transmitted, the feedback reply on block 1 says error, so block 1 is retransmitted next. Meanwhile, block 2 got through, so transmit block 6, etc. The error blocks in feedback replies are x-ed. Just after the second retransmission of block 6, retransmission time $NT = 0$. So

(i) Deliver blocks 8 7 6 5 4 3 2 1 to USER *and* write a copy on CTL.

(ii) Old Data Buffer gets blocks 6 8 7 4; block 4 being the oldest block while 6 is the newest; block 3 having just been confirmed error-free.

Meanwhile, blocks 9, 10, 11, and 12 are being stored in the New Data Buffer. Here we assume the data rate to be 90%, which makes every tenth block a filler. The next block to be transmitted is block 9 and, the 10th being a filler, we use the space to retransmit block 4 for the Clean Tape Log. When block 14 is being transmitted, the status as known to the transmitter is:

ND	CTL	PB
18 17 16 15	9 8 7 6 5 4 3 2 1	14 13 12 11 4
ODB = Empty		

USER
9 8 7 6 5 4_x 3 2 1

Instead of retransmitting block 4 during filler time 10, we may elect to send the new block 11, which is already waiting in ND, since priority is not on retransmissions to the Clean Tape Log. Indeed, this is what should be done to clear the new data buildup in ND and catch up with normal new data flow at the buffer. The subsequent filler block times before the next burst can then be used for retransmissions from the ODB.

III. Simulation

The functional diagram of the simulation program is presented in Fig. 2. The important parameters are NT (the number of retransmissions that can be done within the waiting time), LD (the loop delay or the number of block storage locations in the Priority Buffer), and NR, an integer describing the input data rate R_D: for every NR transmitted block times, $NR - 1$ data blocks enter the system at times $N \neq 0 \bmod NR$ (see Ref. 3). In other words, every NR^{th} transmitted block is a filler for line synchronization. For example, data rates 0.9 and 0.95 data blocks per channel block correspond to NR = 10 and 20, respectively.

The channel errors were generated according to the GCF model developed earlier (Ref. 5) driven with pseudorandom inputs. Block errors were generated directly in the following way: In the model p_i, c_i, $i = 1, \cdots, 4$, are, respectively, the proportion of times spent in the good (error-free) states and the probabilities of entering these states. Then it can be shown that the conditional probability, starting from an error block, of getting a good block *after not more than n error blocks* is given by

$$1 - \left\{ \frac{\Sigma\, ci \dfrac{(1 - p_i^s)^2}{p_i (1 - p_i)}}{\Sigma\, ci \dfrac{(1 - p_i^s)}{p_i (1 - p_i)}} \right\}^{n-1} \tag{1}$$

Similarly, the conditional probability, starting from a good block, of getting an error block *after not more than n good blocks* is given by

$$\frac{\Sigma\, c_i p_i^s \dfrac{(1 - p_i^{s(n+1)})}{p_i (1 - p_i)}}{\Sigma \dfrac{c_i p_i^s}{p_i (1 - p_i)}} \tag{2}$$

It may be noted that (1) and (2) are just the cumulative probabilities of the time, or the number of blocks, until a change of state (error or error-free); the block length is NASA-standard $s = 1200$ bits. Uniformly distributed random numbers between 0 and 1 were generated and used to find the values of n until the first error occurs in (2). By using (2) and (1) alternately in this way with the random numbers, error sequences for the different error phases of the GCF were generated. Each run was continued until 2000 block errors occurred. The experiment consists of counting the number of the 2000 block errors left uncorrected (unretransmitted) within the allowable waiting time t seconds and different loop delay times LD corresponding to the two different loops now being employed on the GCF. The total block lengths of each of the runs varied from 152×10^3 to 190×10^5 in both the high-speed and the wideband data lines. This variation is equivalent to from 8 hours to a few hundred hours of real-time GCF channel use.

IV. Results

A general conclusion from this simulation study is that this scheme can reduce the real-time USER error by at least two orders of magnitude. This error reduction can be achieved if the allowable waiting time is at least $2 \times LD$. This means, for example, that if the longer two-hop link is used between stations, at 4800 bps ($LD = 8$), the user must allow up to 4 seconds for possible retransmissions. If, on the other hand, the shorter one-hop link is used ($LD = 5$), a maximum waiting time of 3 seconds is enough.

The USER error rate decreases with LD. This is so because for short loop delay the transmitter is told earlier about the beginning of a burst and starts retransmitting those blocks already affected, thus preventing further new data from being garbled. Of the 2000 errors in each of the over 300 runs, the maximum USER error was 128 at a waiting time of 3 seconds; more than 50% of the runs delivered less than 10 block errors each to the USER. The error rates for the red, amber, and green error phases of the high-speed data line and the overall average performance in both the high-speed and the wideband modes are shown in Tables 1 and 2 for maximum waiting times varying from 3 to 10 seconds.

The real-time USER error rate is independent of data rate. However, straightforward analysis shows that both the New Data and Old Data Buffer buildups depend strongly on the channel statistics and data rate. A 90% (or 95%) data rate provided enough filler block times to clean up both buffers but the question of how large ND ever gets before it is cleaned up during the long error-free periods is still under investigation. The Receiver Buffer can never be as large as $LD + NT$ but can be as small as LD during repeated retransmissions of consecutive errors.

References

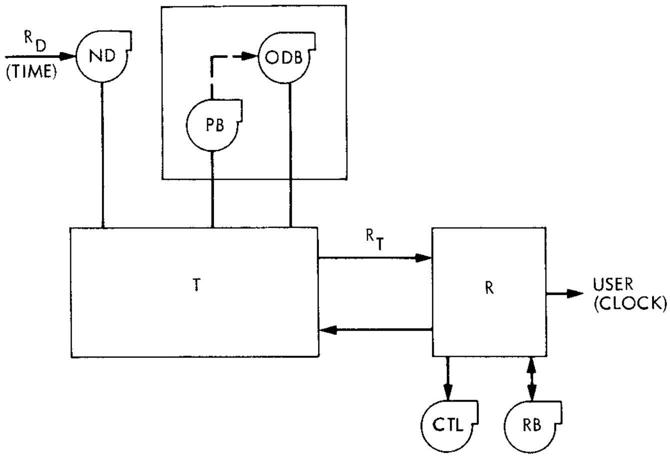
1. Adeyemi, O. H., and McEliece, R. J., "Forward Error Correction for the Ground Communications Facility?" in *The Deep Space Network Progress Report 42-22*, pp. 114–117, Jet Propulsion Laboratory, Pasadena, Calif., Aug. 15, 1974.
2. McClure, J. P., "Basic Algorithm for Retransmission Error Correction," Interoffice Memorandum, Oct. 1972 (JPL internal document).
3. Welch, L. R., "Simulation Study of a GCF Retransmission Scheme," in *The Deep Space Network Progress Report 42-22*, pp. 125–128, Jet Propulsion Laboratory, Pasadena, Calif., Aug. 15, 1974.
4. Adeyemi, O. H., "A Simple GCF Feedback Protocol," IOM 331-75-12, Feb. 1975 (JPL internal document).
5. Adeyemi, O., *Error Control in the GCF: An Information-Theoretic Model for Error Analysis and Coding*, Technical Memorandum 33-699, Jet Propulsion Laboratory, Pasadena, Calif., Oct. 15, 1974.

Table 1. Performance on high-speed data line

Raw block rate (BER)	Maximum USER waiting time, seconds (blocks)	Real-time run duration, hours	USER block rate (BER)
	$LD = 5$		
Red 6.50×10^{-3} (2.45×10^{-4})	3 (12)	29	1.26×10^{-4} $(\times 10^{-6})$
	5 (20)	28.50	5.36×10^{-5} $(\times 10^{-6})$
	6 (24)	28	3.92×10^{-5}
	9 (36)	29	3.45×10^{-5}
	10 (40)	29	3.38×10^{-5}
Amber 2.12×10^{-3} (2.93×10^{-5})	3	82	2.36×10^{-5} $(< \times 10^{-6})$
	5	82.6	1.78×10^{-5}
	6	81	1.38×10^{-5}
	9	84	1.23×10^{-5}
	10	82.5	1.18×10^{-5}
Green 1.18×10^{-3} (3.32×10^{-6})	3	118	4.12×10^{-6}
	5	119.30	3.66×10^{-6}
	6	119.56	2.90×10^{-6}
	9	118	2.35×10^{-6}
	10	116	2.29×10^{-6}
	$LD = 8$		
Red 6.50×10^{-3} (2.45×10^{-4})	3 (12)	27	7.15×10^{-4}
	5 (20)	26.80	6.86×10^{-5}
	6 (24)	25	3.92×10^{-5}
	9 (36)	28	3.81×10^{-5}
	10 (40)	29	3.61×10^{-5}
Amber 2.12×10^{-3} (2.93×10^{-5})	3	82.5	2.95×10^{-5}
	5	82.8	1.93×10^{-5}
	6	79.4	1.66×10^{-5}
	9	82.5	1.32×10^{-5}
	10	82.5	1.21×10^{-5}
Green 1.18×10^{-3} (3.32×10^{-6})	3	116	5.38×10^{-6}
	5	119.4	4.73×10^{-6}
	6	119.5	4.31×10^{-6}
	9	118	3.29×10^{-6}
	10	118	3.06×10^{-6}

Table 2. Overall average performance

Raw block rate (BER)	Maximum USER waiting time, seconds (blocks)	Real-time run duration, hours	USER block rate (BER)
	HSP data line		
	3 (12)	69	2.31×10^{-5} ($\times 10^{-7}$)
$LD = 5$	5 (20)	64.4	2.05×10^{-5}
2.19×10^{-3}	6 (24)	67.7	1.76×10^{-5}
(4.38×10^{-5})	9 (36)	68	1.25×10^{-5}
	10 (40)	65	1.19×10^{-5}
	3	65.25	6.28×10^{-5}
	5	67.4	3.81×10^{-5}
$LD = 8$	6	67.50	2.26×10^{-5}
	9	69	1.76×10^{-5}
	10	69	1.4×10^{-5}
	50-kbps wideband data line		
	3 (125)	8.6	4.35×10^{-5} ($\sim \times 10^{-7}$)
$LD = 31$	5 (209)	8.2	3.00×10^{-5}
1.63×10^{-3}	6 (250)	8	1.39×10^{-5}
(3.54×10^{-5})	9 (375)	7.8	1.12×10^{-5}
	10 (417)	8	1.07×10^{-5}
	3	8.5	5.33×10^{-5}
	5	8.1	4.47×10^{-5}
$LD = 51$	6	8.7	3.52×10^{-5}
	9	8.5	2.32×10^{-5}
	10	8	1.71×10^{-5}



R_D DATA RATE

R_T TRANSMISSION RATE IN BITS PER SECOND; USUALLY $R_D < R_T$, i.e., $R_D = \gamma R_T$, WHERE $0 < \gamma < 1$ MAY BE 90 OR 95%

ND NEW DATA BUFFER. ND HOLDS THE INCOMING NEW DATA STREAM UNTIL IT IS CALLED FOR TRANSMISSION

PB PRIORITY BUFFER

ODB OLD DATA BUFFER

T TRANSMITTER

CTL CLEAN TAPE LOG ON WHICH THE CLEAN DATA ARE WRITTEN AFTER ERROR CORRECTION

RB RECEIVE BUFFER

R RECEIVER

Fig. 1. Retransmission scheme

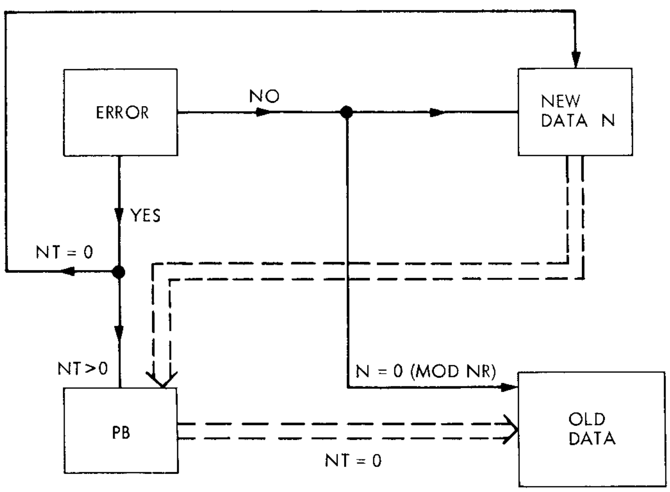


Fig. 2. Simulation functional diagram