Performance of a cyclic redundancy check and its interaction with a data scrambler

by P. E. Boudreau
W. C. Bergman
D. R. Irvin

This paper covers four topics: 1) the operation and performance of cyclic redundancy checks (CRCs); 2) the shortest error patterns of various weights that are undetectable by the ANSI/IEEE-standard 32-bit CRC (CRC32); 3) the general interaction of data scramblers with CRCs; and 4) the specific problems that arise in ATM communication due to the interaction of the scrambler with the degree-10 CRC polynomial (CRC10). Elaborating 4), we explore the virtues of replacing CRC10 with CRC32 or with a degree-10 polynomial (P2055) that has no factors in common with the scrambler. Extensive results are presented concerning the capability of CRC10, P2055, and CRC32 to detect various error patterns.

Introduction and overview

This paper offers

- A tutorial on the operation and performance of cyclic redundancy checks (CRCs).
- New information on the limitations of the ANSI/IEEE-standard 32-bit CRC, specifically on the shortest undetectable error patterns of various weights.
- A tutorial on the interaction of data scramblers and CRCs.
- An extensive analysis of the interaction of the scrambler and CRC proposed for asynchronous transfer mode (ATM) communications.

The first tutorial begins by showing how the working of a CRC can be described as a sequence of operations on polynomials that have modulo-2 coefficients, and how this structure can be extended to cover the problem of undetected data-transmission errors. We then offer a set of definitions that support our understanding of a CRC's capabilities to detect errors, followed by a set of theorems or rules for analyzing its performance.

Attention then turns to the performance of a specific CRC, the 32-bit ANSI/IEEE Standard (CRC32). We apply the rules given in the tutorial to deduce the capability of CRC32 to detect transmission errors. Next, we present new and original results concerning the specific error patterns of weights 2–15 that CRC32 fails to detect.

Returning to the tutorial role, we discuss the general interaction of data scramblers and CRCs, showing that the capability of the CRC to detect errors is reduced whenever its generator polynomial has factors in common with the scrambler polynomial.

Finally, we look closely at specific problems in ATM communication caused by the interaction of the proposed scrambler with the proposed degree-10 CRC (CRC10). These problems are traced to the presence of a common factor held by both the scrambler and the generator polynomials. Consequently, we explore the virtues of choosing a different degree-10 CRC polynomial (called here P2055) that has no factors in common with the scrambler; we also look into the virtues of choosing a degree-32 CRC (CRC32) instead of a degree-10. Extensive results are given concerning the capability of CRC10, P2055, and CRC32 to detect various error patterns. These results suggest that P2055 is a better choice than CRC10—by some measures—and, perhaps more importantly, that CRC32 would clearly be a better choice than any degree-10 candidate for links on which multiple-bit errors might be expected.

Operation of the CRC

The mathematical foundation of the cyclic redundancy check (CRC) is the division algorithm for the ring of polynomials with coefficients taken from an algebraic field: For any two polynomials $M(x)$ and $G(X)$, there exist unique polynomials $Q(x)$ and $R(x)$ such that

- $M(x) = Q(x)G(x) + R(x)$, wherein
- the degree of $R(x)$ is less than the degree of $G(x)$.

Polynomial $Q(x)$ is called the *quotient* and $R(x)$ is called the *remainder*; in the communication problem, $M(x)$ is called the *message*, and $G(x)$ is called the *generator*.

The transmitted sequence, called here $T(x)$, is formed by adding the message and the remainder, thereby making the transmitted sequence a multiple of the generator:

$$T(x) = M(x) + R(x) = Q(x)G(x) + R(x) + R(x).$$

In the communication problem, the polynomials' coefficients are normally drawn from the field of integers modulo-2, making the abstract addition equivalent to the logical *exclusive or*. Consequently,

$$R(x) + R(x) = 0;$$

hence,

$$T(x) = M(x) + R(x) = Q(x)G(x),$$

which shows that the transmitted sequence is a multiple of the generator polynomial. When the transmitted sequence arrives at its destination, the receiver checks to see whether the sequence is still a multiple of the generator polynomial—if it is not, the transmitted sequence has been corrupted in transit by bit errors.

- *The problem of undetected errors*

The pattern of errors impressed on the transmitted sequence can be represented as another polynomial, called here $E(x)$, which is added modulo-2 to $T(x)$ in order to represent the effects of the channel. In this representation, the transmitted sequence experiences bit errors in the positions where $E(x)$ has unit coefficients; so, rather than the transmitted sequence $T(x)$, the receiver actually sees $T(x) + E(x)$.

Suppose that some of the bits of the transmitted sequence are corrupted in transit by channel impairments; i.e., the error polynomial $E(x)$ is nonzero. The remainder found when dividing $T(x) + E(x)$ by the generator $G(x)$ will be nonzero—and the presence of transmission errors will be detected—unless $E(x)$ is a nonzero multiple of the generator $G(x)$. If $E(x)$ is a nonzero multiple of $G(x)$, the remainder computed at the receiver will be zero, and the presence of the transmission errors will not be detected. In other words, error patterns that are multiples of the generator polynomial cannot be detected, but all other error patterns can.

Performance of a CRC

Whether a CRC generated by a certain polynomial is capable of detecting a particular error pattern depends on the factoring relationship of the generator and the errors, as described above. Consequently, a study of the performance capabilities of CRCs can be based on a study of the algebraic characteristics of generator polynomials and their multiples. That is the approach taken here. The remainder of this section defines terms that pertain to the study of such polynomials, and gives some relevant theorems and their practical consequences; mathematical proofs of the theorems are not included [1].

- *Definitions*

- The *weight* of an error pattern $E(x)$ is the number of terms in $E(x)$ that have nonzero coefficients: i.e., the number of *ones* in its bit-sequence representation, or the number of bit errors that it represents.
- The *length of an error pattern* is the number of bits between the first bit in error and the last bit in error, plus two (i.e., the count includes the first and the last bits in error, as well as the number of places between them).
- A polynomial $P(x)$ of degree k is *primitive* if $n = 2^k - 1$ is the smallest value of n for which $P(x)$ is a factor of the polynomial $x^n + 1$. Peterson [2] gives a table of primitive polynomials through degree 34.

- The number $2^k - 1$ is the *natural length* of a degree-k primitive polynomial.
- The *Hamming weight* of a degree-k primitive polynomial is the minimum weight in the set of error patterns that are undetectable by this generator in messages having length less than $2^k - 1$ bits. In other words, the Hamming weight is the weight of the "lightest" undetectable error pattern for messages of less than a specified length, or the minimum number of errors that can go undetected. Thus, a CRC that is based on a primitive generator with Hamming weight h detects any pattern having fewer than h bits in error, but fails to detect at least one error pattern having exactly h bits in error. Note that the Hamming weight of the generator may change as the length of the message changes.
- An *irreducible* polynomial has no nontrivial factors. All primitive polynomials are irreducible, but not all irreducible polynomials are primitive.
- A polynomial $P(x)$ *belongs to exponent* e if e is the least positive integer such that $P(x)$ is a factor of $(x^e + 1)$; the maximum possible value of e is $2^k - 1$, where k is the degree of $P(x)$; the polynomial $P(x)$ is primitive if and only if $e = 2^k - 1$.

- *Theorems*

1. Any one-bit error is detected by a CRC whose generator has more than one term.
2. Any one-bit error or two-bit error pattern of length not exceeding e will be detected by a CRC whose generator belongs to exponent e.
3. If degree-k polynomial $P(x)$ is a factor of $x^e + 1$ for $e < 2^k - 1$, then e is a factor of $2^k - 1$; furthermore, if $P(x)$ belongs to exponent $e < 2^k - 1$, then e is a factor of $2^k - 1$.
4. Any two-bit error pattern of length not exceeding $2^k - 1$ will be detected by a CRC having a primitive degree-k generator. For this reason, a degree-k primitive polynomial generates a CRC that is said to provide *maximal protection* against two-bit errors.
5. The two-bit error pattern $E(x) = x^{2^k-1} + 1$ is undetected by any CRC with a degree-k generator. This follows from a deeper theorem which states that $x^{2^k-1} + 1$ is a multiple of any degree-k generator polynomial $G(x)$. In other words, if $G(x)$ has degree k, a polynomial $Q(x)$ always exists such that $x^{2^k-1} + 1 = Q(x)G(x)$.
6. Any error pattern having a length k or less will be detected by a CRC with a degree-k generator, because a multiple of the degree-k generator cannot have length k or less.
7. Of the set of error patterns having length $k + 1$, the fraction $1/2^{k-1}$ is undetectable by a CRC with a degree-k generator; i.e., this set of error patterns has 2^{k-1} members, one of which is the generator, which corresponds to an undetectable error pattern.
8. Of the set of error patterns having length greater than $k + 1$, the fraction $1/2^k$ is undetectable by a CRC with a degree-k generator. If all error patterns are equally probable, the fraction represents the probability of undetected error. Note that this qualification is more than pedantic, as bit errors in the presence of most scramblers and some modulation techniques [3] cannot be treated as independent events, thereby making some error patterns more likely than others.
9. Any error pattern with an odd weight will be detected by a CRC based on a generator that has $x + 1$ as a factor; a generator with this factor, however, cannot be primitive (it is not irreducible).

Properties of the ANSI/IEEE-standard 32-bit CRC

The ANSI/IEEE-standard 32-bit CRC [4] is generated by the degree-32 primitive polynomial listed by Peterson [2, p. 270] as octal 40460216667:

$$G_{32}(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1.$$

A CRC with generator $G_{32}(x)$ will detect

- Any one- or two-bit error in a message whose length does not exceed $2^{32} - 1$ bits, because $G_{32}(x)$ is primitive.
- Any error pattern having length less than 33 bits.
- All but the fraction $1/2^{31}$ of possible error patterns having a length of 33 bits (i.e., only one 33-bit pattern will be undetected—the error pattern corresponding exactly to CRC_{32}).
- All but the fraction $1/2^{32}$ of possible error patterns having a length greater than 33 bits.

Table 1 gives the shortest undetectable error patterns of weights between two and fifteen.* In this table, an error pattern is represented by the exponents of its terms that have nonzero coefficients; for example, the weight-4 error pattern $x^{3006} + x^{2866} + x^{2215} + 1$ is listed as (3006, 2866, 2215, 0). Jain [5] has proposed a subset of our Table 1 for weights three through thirteen. The additional information we present in Table 1 for weights thirteen through fifteen completes the picture; moreover, this new information shows that the patterns for weights thirteen and fourteen are not unique, whereas the patterns for the lower weights might be.

*This list corrects errors in an earlier work: contribution IEEE 802.85*1.198, June 25, 1985; the author of that work agrees with our corrections.

Table 1 Shortest undetectable error patterns for ANSI/IEEE-standard 32-bit CRC.

Weight	*Shortest undetectable error pattern*	*Unique?*
2	4294967295, 0	yes
3	91639, 41678, 0	yes
4	3006, 2866, 2215, 0	yes
5	300, 155, 117, 89, 0	not known
6	203, 186, 123, 85, 79, 0	not known
7	123, 120, 80, 74, 53, 45, 0	not known
8	89, 88, 41, 36, 16, 13, 5, 0	not known
9	66, 57, 37, 32, 19, 18, 3, 2, 0	yes
10	53, 38, 36, 33, 30, 27, 25, 7, 3, 0	yes
11	44, 43, 41, 37, 35, 32, 31, 16, 7, 3, 0	yes
12	42, 30, 26, 24, 21, 18, 13, 8, 7, 5, 3, 0	yes
13	42, 40, 37, 35, 33, 29, 23, 20, 18, 15, 6, 1, 0	1 of 2
13	42, 41, 40, 34, 32, 29, 19, 17, 12, 10, 8, 4, 0	1 of 2
14	42, 39, 29, 27, 26, 21, 19, 15, 14, 8, 6, 2, 1, 0	1 of 3
14	42, 40, 37, 34, 28, 27, 26, 20, 19, 17, 13, 6, 3, 0	1 of 3
14	42, 36, 34, 28, 27, 25, 21, 20, 16, 15, 11, 10, 2, 0	1 of 3
15	32, 26, 23, 22, 16, 12, 11, 10, 8, 7, 5, 4, 2, 1, 0	yes*

*This is the generator CRC32 itself.

Interaction of a scrambler and a CRC

To allow the receiver to derive its clock in the presence of long runs of binary ones or zeros, some modulation techniques require that transmitted data be passed through a scrambler. A scrambler works—in abstraction—much like a CRC; i.e., the working of the scrambler can be described by a sequence of operations on finite-field polynomials.

Let the character of the scrambler be defined by the polynomial $S(x)$. In the presence of transmission errors processed by the scrambler, the receiver sees the bit sequence

$$[T(x) + E(x)]S(x) = T(x)S(x) + E(x)S(x).$$

The remainder found by computing the CRC of the term $T(x)S(x)$ is always zero, because $G(x)$ is a factor of $T(x)$, and therefore a factor of the product $T(x)S(x)$. Consequently, the effects of the channel impairment are described by the other term, called here $F(x)$:

$$F(x) = E(x)S(x).$$

The designation F was chosen for its mnemonic value in suggesting *false* errors, which are those generated by the actions of the scrambler over and above those errors represented by $E(x)$, which are called *line errors*. Some false errors may occur within the bounds of $T(x)$. Others may fall outside the bounds of $T(x)$; it is convenient and descriptive to say that such errors *spill out of the message*. The presence of errors that spill out of the message should be expected, because the degree of $F(x)$ can exceed the degree of $T(x)$ by as much as the degree of the scrambler polynomial.

Beyond the effects of false errors introduced by the scrambler, the presence of the scrambler may weaken the intrinsic performance of the CRC. If $S(x)$ is a multiple of the generator polynomial $G(x)$, the false-error term $F(x)$ will have $G(x)$ as a factor, leading to an always-zero remainder when the CRC of $F(x)$ is computed at the receiver, thereby nullifying the capability of the CRC to detect errors. In a less catastrophic way, the power of the CRC is weakened whenever $G(x)$ and $S(x)$ have common factors.

Errors that spill out can simply be ignored if the scrambler is restarted with each new message. In this situation, the full capability of the CRC should be maintained when the scrambler and the generator have no common factors; for example, an irreducible polynomial could be chosen as the scrambler. If the scrambler is allowed to run continuously, however, the effects of spilled errors must be considered in any analysis of the CRC's performance. Moreover, no simple rules can be given for understanding the scrambler's impact on the capability of the CRC—each case must be examined on its own.

ATM communication and its scrambler

ATM (asynchronous transfer mode) communication is based on the segmentation of a data frame into a plurality of 53-byte cells, where these cells each carry a 48-byte payload (384 bits) and have a five-byte header. The payload CRC proposed for ATM Adaptation Layer type 3/4 (AAL-3/4) provides a 10-bit CRC generated by a polynomial called CRC10,

$$\text{CRC10} = x^{10} + x^9 + x^5 + x^4 + x + 1,$$

as described more fully by Dravida and Damodaram [6]; Simmons and Gallager [7] discuss the specifics of the ATM error-control problem in more detail.

The ATM bit stream is scrambled according to the polynomial $S(x) = x^{43} + 1$; during the transmission of the five-byte headers, however, the scrambler is turned off, but not reset. Moreover, a number of sources may be multiplexed, meaning that cells that are adjacent in the ATM stream may not belong to the same source–destination pair.

To accommodate the effects of the scrambler and multiplexer, the ATM error-control mechanism must be capable of detecting several kinds of errors:

- Errors within the received sequence that are caused by impairments to the transmission channel.
- Errors spilling into the received sequence as the result of the scrambler's operations on transmission errors that occurred in the last 43 bits of the previous cell.
- Errors spilling into the received sequence as the result of the scrambler's operations on transmission errors that occurred in the first 341 bits of the cell itself.

Accordingly, the performance of CRC10 is to be judged against the following three criteria:

1. All error patterns resulting from one- or two-bit line errors must be detected in the case wherein no false errors spill into the cell's 384-bit payload.
2. All error patterns must be detected in the case wherein the cell experiences a one-bit line error in the last 43 bits of the payload (meaning that the scrambler's image of this error spills into the next cell) and a one-bit false error spills into the cell as the result of the scrambler's actions in the previous cell.
3. All error patterns must be detected in the case wherein 1) a single line error occurs, 2) the scrambler's image of that error spills into the same cell, and 3) a one-bit false error spills into the cell as the result of the scrambler's actions in the previous cell.

- *The capability of CRC10 with the ATM scrambler*

The generator polynomial CRC10 can be factored into the product of two primitives:

$$\begin{aligned}\text{CRC10} &= x^{10} + x^9 + x^5 + x^4 + x + 1 \\ &= (x + 1)(x^9 + x^4 + 1).\end{aligned}$$

By the theorems given earlier, the primitive term $(x + 1)$ should detect all error patterns with odd weight (i.e., an odd number of bit errors) as seen by the CRC decoder; furthermore, the primitive term $(x^9 + x^4 + 1)$ should detect all two-bit error patterns as seen by the CRC decoder, provided that the message length does not exceed 511 bits, which is always the case in the ATM problem. The caveat *as seen by the CRC decoder* is a reminder that the error pattern seen by the CRC decoder differs from the line-error pattern, thanks to effects of the scrambler.

The scrambler polynomial can be factored into four irreducible terms:

$$S(x) = (x^{43} + 1) = \prod_{i=1}^{4} S_i(x),$$

where

$$\begin{aligned}S_1(x) &= (x + 1), \\ S_2(x) &= (x^{14} + x^{13} + x^{11} + x^7 + x^3 + x + 1), \\ S_3(x) &= (x^{14} + x^{12} + x^{10} + x^7 + x^4 + x^2 + 1), \\ S_4(x) &= (x^{14} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 \\ &\quad + x^4 + x^3 + 1).\end{aligned}$$

Because the term $(x + 1)$ is common to the generator and the scrambler, the error-detecting capabilities of this term are nullified, thereby annulling CRC10's claim to detecting all error patterns with an odd weight. Thus, in the presence of the scrambler, the capability of CRC10 is reduced to the capability of the remaining factor $(x^9 + x^4 + 1)$. The question naturally arises: "Can we find a degree-10 generator polynomial that works better than CRC10 with the $(x^{43} + 1)$ scrambler?"

- *Other 10-bit candidates for generating the ATM CRC*

In the search for a new CRC generator, we can eliminate the set of reducible polynomials from further consideration by the following argument: The theorems given earlier show that the generator needs a degree-9 (or higher) factor to ensure the detection of all two-bit errors in a 384-bit message. Furthermore, we disallow x as a factor of the generator to ensure the detection of all single-bit errors. This leaves $(x + 1)$ as the only other factor suitable for multiplying a degree-9 factor to construct a degree-10 generator. Since $(x + 1)$ is a factor of the scrambler, we disallow it as a factor of the new generator; indeed, the desire to avoid this factor motivates the search for a new generator. Consequently, a suitable generator that is a reducible polynomial cannot be found.

The remaining polynomials—those that are irreducible—can be divided into two categories: polynomials that are primitive, and polynomials that are not primitive. By the following argument, we can eliminate the nonprimitives from further consideration: The largest factor of $2^{10} - 1$ is 341. Consequently, the largest exponent to which a degree-10 nonprimitive can belong is 341. Consequently, the detection of two-bit errors by a degree-10 (nonprimitive) polynomial is ensured only for messages whose lengths

do not exceed 341 bits. In this problem, however, the message length is 384 bits. We therefore eliminate degree-10 polynomials that are not primitive as candidates to replace CRC10, leaving only the set of degree-10 primitive polynomials for consideration.

Sixty degree-10 primitive polynomials are identified by Peterson [2]. We have examined several of these exhaustively, recreating all possible error patterns by computer simulation and checking for undetectable errors, thereby verifying that the several polynomials indeed meet the three points of the judgement criteria *in the particular circumstances of this problem*. In more general circumstances, however, there is no guarantee that a degree-10 primitive can detect all three-bit error patterns, which is a limitation that (inappropriately) suggests that a degree-10 primitive could not meet point 3 of the judgement criteria.

Among the degree-10 polynomials that meet the criteria established for this problem, one would seem *a priori* about as good as another. We therefore arbitrarily select the polynomial P2055 = $x^{10} + x^5 + x^3 + x^2 + 1$ for further study. In octal notation, this is the polynomial 2055.

• *Comparing CRC10, P2055, and CRC32*

Table 2 shows the error-detecting capability of three generators—CRC10, P2055, and CRC32—under various permutations of the numbers of line errors, errors that spill into a cell, and errors that spill out; these permutations are called *cases* as a matter of convenience. The results shown in Table 2 were found by appealing to the theorems given earlier, or, for cases beyond the reach of the theorems, by examining the remainders $R(x)$ associated with all possible error patterns with the help of a computer, checking for undetectable errors. This exhaustive examination exploited the linearity of the CRC: The remainder associated with a particular error pattern was found by summing (modulo-2) the remainders of the error pattern's individual terms. For example, the remainder associated with the error pattern $x^3 + x^2 + 1$ was found by summing the remainder associated with x^3, the remainder associated with x^2, and the remainder associated with x^0 or 1. In this context, the remainders are often called "syndromes."

In Table 2, the cases are numbered 1 through 45 in the first column; the nature of each case is given in columns two through five; the performance of CRC10, P2055, and CRC32 is given in columns six, seven, and eight, respectively. The entries in columns six through eight are counts of the numbers of different error patterns undetected by the CRCs identified by the columns' headers. A count is listed simply as "F" wherever at least one undetected error pattern was found, but where the precise number of undetected patterns could not be determined.

• *Discussion of Table 2*

Earlier, we listed the criteria for judging the performance of a CRC in the context of this problem; these criteria are repeated here for the sake of convenience. In the problem at hand, the performance of CRC10 is to be judged against the following three points:

1. All error patterns resulting from one- or two-bit line errors must be detected in the case wherein no false errors spill into the cell's 384-bit payload.
2. All error patterns must be detected in the case wherein the cell experiences a one-bit line error in the last 43 bits of the payload (meaning that the scrambler's image of this error spills into the next cell) and a one-bit false error spills into the cell as the result of the scrambler's actions in the previous cell.
3. All error patterns must be detected in the case wherein 1) a single line error occurs, 2) the scrambler's image of that error spills into the same cell, and 3) a one-bit false error spills into the cell as the result of the scrambler's actions in the previous cell.

All three of the generators examined here meet the points of the criteria. Within Table 2, entries (2) through (6) show that point 1 is met; entry (13) shows that point 2 is met; entry (12) shows that point 3 is met.

Further, according to the entries of Table 2, at least three line errors must occur—sometimes spread over two adjacent cells given the presence of the scrambler—in order to produce an error pattern that is undetected by CRC10 or P2055. The appearance of an asterisk (*) in Table 2 denotes the relevant entries. Because three line errors are required to generate these patterns, the probability of occurrence of the associated undetected-error pattern is proportional to p^3, where p is the bit-error ratio of the underlying transmission medium, here assuming that line errors can be modeled as independently occurring events. Accounting for the occurrence of higher-order error patterns in the same way suggests that the probability of any undetected error is given by the sum of a finite sequence of terms in p^N, where $N \geq 3$. For small values of p, this sum is dominated by the p^3 term.

Without the scrambler and the resulting loss of $(x + 1)$ as a factor of CRC10, however, all three-bit error patterns would be detected by CRC10, and the sequence for the probability of any undetected error would be dominated by the p^4 term. Thus, the effect of the scrambler on CRC10 is to weaken the performance in allowing undetected errors by (approximately) the factor p.

Other entries in Table 2 suggest that P2055 is superior to CRC10 in detecting even numbers of errors, while CRC10 is superior to P2055 in detecting odd numbers. Entries (41) through (45), however, suggest that neither CRC10 nor P2055 is very good at protecting against multiple errors—

CRC32 is quite superior to either of the degree-10 choices, as would be expected. Note from Table 2 that five or more bit errors must occur before the possibility of errors undetected by CRC32 is opened. According to the same kind of argument constructed above, the probability of undetected errors with CRC32 and the scrambler is dominated at worst by the p^5 term of the summed sequence. Thus, replacing CRC10 with CRC32 when the scrambler is present would restore (and surpass) the performance offered by CRC10 absent the scrambler.

- *Further observations*

The following behavior was observed for CRC10 and P2055 in the presence of the $(x^{43} + 1)$ scrambler:

- The length of the shortest undetected three-bit line error is
  - CRC10—ten bits, exponents (0, 4, 9), occurring 332 times in 384 bits.
  - P2055—twenty bits, exponents (332, 341, 351), occurring nine times in 384 bits.
- The length of the most frequently occurring three-bit undetected error pattern is
  - CRC10—ten bits, exponents (0, 4, 9), occurring 332 times in 384 bits.
  - P2055—56 bits, exponents (0, 28, 55), occurring 286 times in 384 bits.

Finally, a consecutive run of error patterns undetected by CRC10 was unexpectedly found. These patterns are created by three errors spilling into the 384-bit cell in positions (0, 16, 36), accompanied by two line errors falling into positions $(0 + N, 370 + N)$, where $0 \leq N \leq 13$.

Concluding remarks

We have given a set of theorems useful in characterizing the error-detecting capabilities of CRC polynomials, a tutorial on the interaction between data scramblers and CRCs, and extensive results on the characteristics and performance capabilities of a number of different polynomials. Throughout, the implicit metric of goodness has been the capability to detect errors. Clearly, this metric would be but one of many in any real-world system-design problem. Other considerations not mentioned here would inevitably include transmission overhead, error-correction capability, implementation complexity, and so forth.

Table 2 Performance comparison of CRC10, P2055, and CRC32.

Case	Number of errors				Generator		
	Spilled		Line error in cell	Total seen by CRC	CRC10	P2055	CRC32
	in	out					
(1)	0	0	0	0	0	0	0
(2)	0	0	1	2	0	0	0
(3)	0	1	1	1	0	0	0
(4)	0	0	2	4	0	0	0
(5)	0	1	2	3	0	0	0
(6)	0	2	2	2	0	0	0
(7)*	0	0	3	6	12,739	6,403	0
(8)*	0	1	3	5	0	2,366	0
(9)*	0	2	3	4	578	437	0
(10)	0	3	3	3	0	0	0
(11)	1	0	0	1	0	0	0
(12)	1	0	1	3	0	0	0
(13)	1	1	1	2	0	0	0
(14)*	1	0	2	5	0	2,639	0
(15)*	1	1	2	4	1,296	550	0
(16)*	1	2	2	3	0	50	0
(17)	1	0	3	7	0	276,030	0
(18)	1	1	3	6	209,610	105,305	0
(19)	1	2	3	5	0	13,013	0
(20)	1	3	3	4	832	542	0
(21)	2	0	0	2	0	0	0
(22)*	2	0	1	4	676	280	0
(23)*	2	1	1	3	0	39	0
(24)	2	0	2	6	101,957	50,319	0
(25)	2	1	2	5	0	13,242	0
(26)	2	2	2	4	1,496	671	0
(27)	2	0	3	8	F	F	?
(28)	2	1	3	7	0	F	?
(29)	2	2	3	6	F	F	?
(30)	2	3	3	5	0	F	?
(31)	3	0	0	3	0	0	0
(32)	3	0	1	5	0	4,236	0
(33)	3	1	1	4	1,123	541	0
(34)	3	0	2	7	0	698,829	?
(35)	3	1	2	6	352,612	177,401	?
(36)	3	2	2	5	0	10,908	?
(37)	3	0	3	9	0	F	?
(38)	3	1	3	8	F	F	?
(39)	3	2	3	7	0	F	?
(40)	3	3	3	6	F	F	?
(41)	0	0	4	8	1,087,011	541,829	0
(42)	0	1	4	7	0	274,910	0
(43)	0	2	4	6	102,099	51,206	0
(44)	0	3	4	5	0	4,132	0
(45)	0	4	4	4	323	100	0

*is a marker for reference later in the text.

Table entries under columns two through five identify the nature of the cases; entries under columns six through eight give the counts of the unidentified error patterns. "F" means that one or more undetected error patterns were observed, but the count of such patterns could not be determined; "?" means that no information is available.

References

1. Proofs of similar theorems are discussed by W. W. Peterson and D. T. Brown, "Cyclic Codes for Error Detection," *Proc. IRE* **49,** 228–235 (January 1961).
2. W. W. Peterson, *Error-Correcting Codes*, The M.I.T. Press, Cambridge, MA, 1961.
3. C. L. Chen and R. A. Rutledge, "Error Correcting Codes for Satellite Communication Channels," *IBM J. Res. Develop.* **20,** 168–175 (March 1976).
4. *Token Ring Access Method and Physical Layer Specification*, ANSI/IEEE Standard 802.5-1985, Institute of Electrical and Electronics Engineers, 345 E. 47th St., New York, 1985.
5. R. Jain, "Error Characteristics of Fiber Distributed Data Interface (FDDI)," *IEEE Trans. Commun.* **38,** No. 8, 1244–1252 (August 1990).

6. S. Dravida and R. Damodaram, "Error Detection and Correction Options for Data Services in B-ISDN," *IEEE J. Sel. Areas in Commun.* **9,** No. 9, 1484–1495 (December 1991).
7. J. Simmons and R. Gallager, "Design of Error Detection Scheme for Class C Service in ATM," *IEEE/ACM Trans. Networking* **2,** No. 1, 80–88 (February 1994).

Received March 18, 1993; accepted for publication May 28, 1994

Paul E. Boudreau *7405 Amaris Lane, Raleigh, North Carolina 27612.* Dr. Boudreau is now an independent consultant in the fields of mathematics and telecommunications, having retired from IBM in 1990 as a Senior Technical Staff Member. He joined IBM's General Products Division, Endicott, New York, in 1958, and in 1965 transferred to Research Triangle Park, North Carolina, to work on a wide range of topics related to telecommunications. In 1972, he moved to the mid-Hudson Valley, where he held executive positions in corporate and divisional headquarters. He returned to the Research Triangle Park in 1978 as manager of IBM's Systems Network Architecture (SNA), and remained at RTP until his retirement, working in the fields of telecommunications and SNA. He has received more than fifteen awards for his work. Dr. Boudreau received the B.S. from the University of Vermont in 1953, the M.S. from the University of Rochester in 1958, and the Ph.D. in mathematics from the University of Michigan in 1965. He is a member of Phi Beta Kappa, Phi Kappa Phi, Kappa Delta Pi, the American Mathematical Society, and the Mathematical Association of America. He has published more than twenty papers, and holds five patents that pertain to the control of transmission errors.

William C. Bergman *IBM Networking Systems, P.O. Box 12195, Research Triangle Park, North Carolina 27709 (BERGMAN at RALVMG).*

David R. Irvin *Department of Electrical and Computer Engineering, North Carolina State University, P.O. Box 7911, Raleigh, North Carolina 27695.* In August 1994, Mr. Irvin joined the Department of Electrical and Computer Engineering at North Carolina State University, where he teaches the principles of electrical engineering. Prior to joining NCSU, he worked twenty years for the IBM Corporation at Research Triangle Park, North Carolina, in the fields of telecommunication analysis, communication and network management architecture, system performance analysis, transmission technology, and digital signal processing. He was twice named IBM–RTP Author of the Year for collected works in these fields, and he has received numerous other IBM awards for his accomplishments. Before joining IBM, he worked as an electrical engineer under contract to the United States Navy. Mr. Irvin received the B.E.S. in 1970 from The Johns Hopkins University, the M.E.E. in 1971 from North Carolina State University, and the P.D.E. in 1990 from the University of Wisconsin–Madison. He is a member of Phi Beta Kappa, Sigma Xi, and Tau Beta Pi, and a Senior Member of the Institute of Electrical and Electronics Engineers. He holds six patents, two of which pertain to cyclic redundancy checks, and has published widely in the professional literature.