Coverage and the Use of Cyclic Redundancy Codes in Ultra-Dependable Systems

Michael Paulitsch* Jennifer Morris† Brendan Hall*
Kevin Driscoll* Elizabeth Latronico† Philip Koopman†
*Honeywell**, Carnegie Mellon University*†
{*michael.paulitsch, brendan.hall, kevin.driscoll*} *@honeywell.com**
{*jenmorris, beth, koopman*} *@cmu.edu*†

Abstract

A Cyclic Redundancy Code (CRC), when used properly, can be an effective and relatively inexpensive method to detect data corruption across communication channels. However, some systems use CRCs in ways that violate common assumptions made in analyzing CRC effectiveness, resulting in an overly optimistic prediction of system dependability. CRCs detect errors with some finite probability, which depends on factors including the strength of the particular code used, the bit-error rate, and the message length being checked. Common assumptions also include a passive network inter-stage, explicit data words, memoryless channels, and random independent symbol errors. In this paper we identify some examples of CRC usage that compromise ultra-dependable system design goals, and recommend alternate ways to improve system dependability via architectural approaches rather than error detection coding approaches.

1. Introduction

Recent industrial trends point to an increased reliance on computerized control in "ultra-dependable" distributed embedded systems. An ultra-dependable system is one whose malfunction results in significant financial loss or loss of human life. A good example of an ultra-dependable distributed embedded system is a commercial aircraft. According to government aviation regulations, commercial aircraft "*must be designed to ensure that they perform their intended functions under any foreseeable operating condition*" [15]. Similar requirements for other ultra-dependable systems can be found in IEC 61508 [22].

A common reliability requirement for existing ultra-dependable systems is 10^{-9} failures/hour. It can be argued that large-scale deployment of ultra-dependable systems, such as in drive-by-wire applications for automobiles, requires even more stringent reliability targets due to increased passenger hour exposure [28]. Even worse, systems with large-scale deployments tend to have significant cost constraints – cars cost about one thousandth as much as large passenger jets.

One method of achieving ultra-dependability is through the use of distribution to tolerate faulty subsystems or spatially correlated faults. A key component of such a distributed embedded system is a dependable communication system. Ultra-dependable communication systems are already common in the aviation domain, and are emerging in the automotive market as well (e.g., so-called X-by-wire systems that include steer-by-wire and brake-by-wire applications). These communication systems often use Cyclic Redundancy Codes (CRCs), which provide relatively inexpensive in-line error detection of data corruption.

This paper reviews the use of CRCs in systems from the viewpoint of ultra-dependable system design. It is the objective of this paper to revisit CRC protection mechanisms with respect to probabilistic strength and underlying assumptions. We identify examples of ultra-dependable systems in which these assumptions do not hold, and present alternate design strategies for achieving adequate error detection coverage. In particular, we focus on the implications and design assumptions of active intermediate communication stages in the network topology and the use of implicit data in CRC calculations.

1.1. Overview of Paper

The rest of this paper is organized as follows: Section 2 gives an overview of ultra-dependable system design. Section 3 provides background on CRCs, including the underlying assumptions of their error detection capabilities. Section 4 provides specific examples of incompatibilities between CRC usage assumptions and applications in ultra-dependable systems. Section 5 presents recommendations for improved ultra-dependable design practices. Finally, section 6 presents our conclusions.

2. Ultra-Dependable Systems

The electronic hardware components from which ultra-dependable "10^{-9} systems" are built have a typical failure probability of 10^{-6} failures/hour. While permanent failure rates of these components have been slightly decreasing during recent years (getting better; 10^{-7} to 10^{-8} failures/hour), the transient failure rates have actually been increasing, and are often up to 100 times more frequent than permanent failures [12]. Using a component failure rate of 10^{-6} seems to be a safe bound ([45] provides guidance for reliability numbers and influencing factors).

Although the failure rate of data source components within a distributed system is on the order of 10^{-6} to 10^{-7} failures/hour, the communication subsystem requires a much lower failure rate (10^{-9} or less), because it serves as the "glue" between all sources and sinks of a distributed architecture. All fault-tolerance mechanisms at higher layers are typically built with an assumption of ultra-dependable communications. In general, complete loss of communications causes a system failure.

Integrity and availability are two attributes of dependability that are especially relevant for dependable communication systems and the analysis performed in this paper. *Integrity* is the *"absence of improper system state alterations"* [4]. With respect to communication systems and CRCs, this means that any communication failure should be detected with sufficiently high probability (i.e., the probability of any undetected failure must be sufficiently low).

Availability is *"readiness for correct service"* [4]. For fail-operational dependable embedded systems, this means that the communication system needs to be available nearly all the time. Moreover, any outage that occurs during system operation must be very short. For example, [20] gives a minimum steer-by-wire system outage of no more than 50 msec during vehicle operation. Limitations in system resources may require a tradeoff between availability and integrity. The art of dependable system design is to ensure both have sufficiently high levels.

It is usually impracticable to test ultra-dependable systems long enough to assure high dependability (Butler and Finelli make this argument for ultra-reliable software [9]). Ultra-dependable system designers must therefore invest a significant amount of time during the development process on design assurance (e.g. DO-254 [38], DO-178B Level A [37], IEC 61508 SIL 3,4 [22]). As a result, the accuracy of the dependability analysis of ultra-dependable systems is extremely important.

3. Cyclic Redundancy Codes

Cyclic redundancy codes (also known as cyclic redundancy checks) have long been used for error detection in computing. [33] and [35] are among the commonly cited standard reference works for CRCs. This paper focuses on the use of CRCs to detect medium-induced errors in messages transferred over communication channels. CRCs are also commonly used to protect the integrity of data stored in memory.

A CRC can be thought of as a (non-secure) digest function for a data word that can be used to detect data corruption. Mathematically, a CRC can be described as treating a binary data word as a polynomial over GF(2) (i.e., with each polynomial coefficient being zero or one) and performing polynomial division by a generator polynomial G(x). The generator polynomial will be called a CRC polynomial for short. (CRC polynomials are also known as feedback polynomials, in reference to the feedback taps of hardware-based shift register implementations.) The remainder of that division operation provides an error detection value that is sent as a Frame Check Sequence (FCS) (also called a syndrome) within a network message or stored as a data integrity check. Whether implemented in hardware or software, the CRC computation takes the form of a bitwise convolution of a data word against a binary version of the CRC polynomial.

Error detection is performed by comparing an FCS computed on a piece of retrieved or received data against the FCS value originally computed and either sent or stored with the original data. An error is declared to have occurred if the stored FCS and computed FCS values are not equal.

Bit error properties are often evaluated using test equipment emulating field characteristics. Evaluating performance in the wide variety of operating circumstances found in embedded systems is difficult in general. In practice CRCs have been found to be an adequate protection mechanism for deployment in everyday communication systems for medium-induced errors. But this experience is insufficient to assure ultra-dependable operation.

3.1. Probabilistic Strength of CRC Polynomials

The error detection probability of CRCs have been studied extensively [10, 25, 27, 1, 7, 17, 32, 18]. The key research topic of interest is the probability of undetected error for CRCs given various fault assumptions.

As with all digital signature schemes, there is a small, but finite, probability that a data corruption will occur that inverts a sufficient number of bits in just the right pattern such that the error is undetectable. The minimum number of bit inversions required to achieve such undetected errors, the Hamming distance, is dependent on the CRC polynomial and the length of the data word.

For a given CRC polynomial, a (binary) data word of length m and an FCS of lengths k there are $\frac{2^{m+k}}{2^k} = 2^m$ correct code words. A k-bit CRC used as an FCS detects all burst errors in the data word up to length k. The probability that a random data word will produce a particular FCS is $\frac{2^m}{2^k}/2^m = 2^{-k}$. Example: For an FCS of 24 bits, the likelihood that a different data word will produce the "correct" FCS is $2^{-24} = 6 \cdot 10^{-8}$. It has been shown that 2^{-k} is a good approximation of the upper bound of undetected errors for most applications; all assuming a uniform error distribution [1, 7, 17, 32].

Note that one cannot assume the associated failure modes of devices will follow any common mathematical distribution. As a result, a component suffering a systematic arbitrary fault might not produce uniformly distributed random messages. If the messages are not uniformly distributed, in the worst case the FCS of the generated message could always match the FCS of the original message, in which case the CRC would provide no protection. In this case, the end-to-end failure rate would equal the component's failure rate.

3.2. Error Model Assumptions

In addition to the bit error rate, other characteristics of the error model play an important role in CRC effectiveness. Common assumptions include explicit data words, uncorrelated errors from bit to bit, memoryless channels, and a passive network interstage. This section discusses how the probabilistic arguments presented in section 3.1 apply to ultra-dependable system design.

3.2.1. Network Medium. In the preceding section, the probability was calculated as if CRC polynomial performance were the sole determining factor of error detection capability, and assumed that all errors encountered resulted in bit inversions that were undetectable by other means. When considering the probability of undetected, medium-induced errors, physical layer influence and other effects have to be considered. These effects have a significant effect on the overall end-to-end undetected error probability.

The choice of encoding rule in the underlying physical layer may improve the error detection probabilities for medium-induced errors. For example, framing errors and errors leading to invalid symbol encodings can be detected and provide a layer of error detection capability before received messages are checked for a valid FCS value. In [26], Koopman investigated the error detection probability of the TCN (Train Communication Network) and its overall error detection capabilities. TCN uses Manchester encoding for the physical channel. Undetectable symbol inversions between zero and one require sigmoid-shaped noise functions, which seem unlikely and for which there is no known phenomenology in train applications. The use of a semi-bit error model that assumed errors would take the form of independent "semi-bit" flips found that most noise-induced data errors would be detected at the bit encoding level. This approach to analysis greatly increased the expected overall error detection performance. Similarly, Stone et al. found for real world tests on an ATM (Asynchronous Transfer Mode) network that certain failures are detected even without CRC or checksum use [41].

The probabilistic strength of CRCs depends on the bit error rate of the network, performing best for single burst errors smaller than the FCS size (which are all detectable) and for individual uncorrelated bit errors. The bit error rate is highly dependent on the network medium, system deployment environment and communication speed. Typical values are 10^{-6} to 10^{-13} errors/bit, although error rates in the field are said to vary dramatically. Some standards for optical networks specify maximum allowable BERs, such as $2.5 * 10^{-10}$ in the Boeing ARINC 636 standard [11] and 10^{-12} for Gigabit Ethernet [40]. Copper BERs are typically higher. For example, the Train Communication Network conformance tests allow only three frame errors in $3 * 10^{6}$ frames, for a BER of about 10^{-6} to 10^{-7} for customary small frame sizes [23]. Field data from a Controller Area Network factory automation application cites a measured BER of approximately $3.8 * 10^{-7}$ [3].

If the uncorrelated bit error assumption holds true, the probabilistic strength of CRCs is generally improved compared to 2^{-k}. In a single uncorrelated bit error model, the corrupted data word is not random, since the corrupted data word is a function of the original data word. To produce an undetected error, the number of bit errors must first equal or exceed the Hamming distance (HD) of the CRC. Therefore, the overall probabilistic strength is 2^{-k} multiplied by the probability that the number of bit errors equals or exceeds the HD. For single uncorrelated bit errors, this probability is approximately (BER * message size)^{HD}, multiplied by the number of messages per hour.

Another common assumption of CRC error detection analysis is the use of memoryless communication channels. A communication channel is called *memoryless* if the noise affects each transmitted symbol independently. Memoryless channels are also called *random-error channels*. On *channels with memory*, the noise is not independent from symbol to symbol. Examples of channels with memory are burst-error channels and fading channels [33, p.9,13].

Kuznetsov et al. present one of few papers that investigate the undetected error probability on channels with memory [29]. Lin and Costello even conclude: "*Appropriate models for channels with memory are difficult to construct, and coding for these channels is more of an art than a science*" [33, p.9].

These examples demonstrate that without knowledge of the possible failure modes of the communication channels and error detection capabilities of the physical encoding, a computed integrity value based on the usual set of CRC performance assumptions will not be an accurate prediction of real-world experience.

3.2.2. Byzantine Errors. While it might seem odd at first sight, it is possible that data can be marked with a correct CRC FCS, but *different* data values are seen at *different* receivers. Driscoll et al. call this undetected error a "Schrödinger's CRC" [14]. Schrödinger's CRC is a Byzantine fault that can occur if selected bits of a message can be interpreted differently at different receivers, i.e. one or more bits of a message can be seen as a zero at some receivers but as a one at other receivers.

One potential cause for this type fault is a weak driver on a bus. A similar scenario is a stuck-at $\frac{1}{2}$ bit in the transmitter. For example, a transceiver might send some weak bits (called $\frac{1}{2}$ bits because the analog voltage level is neither logical one nor logical zero) to several receivers. As the threshold levels are likely different at different receivers, $\frac{1}{2}$

bits can be interpreted differently. If the number of $\frac{1}{2}$ bits is larger than the HD, a receiver might receive a corrupted message with a valid CRC-based FCS, but incorrect data.

The probability of a Schrödinger's CRC is hard to evaluate. A worst-case estimate of its occurrence due to a single device is the device failure rate. This phenomenon is a source error, but it has some resemblance to inter-stage failures (Section 4.2) due to a systematic failure of a single device, described in a subsequent section.

Powell et alii describe message authentication in the contents Byzantine failures and their coverage as well as traditional solutions in [36].

4. Questionable CRC Use In Ultra-Dependable Systems

Section 3.2 described assumptions of the CRC that influence the end-to-end error-detection coverage. This section presents examples of systems in which those assumptions might be violated, thereby potentially decreasing the end-to-end error-detection coverage of the CRCs.

4.1. Implicit Data Words

Some communication protocols compute the CRC frame check sequence over several fields but send only some fields as part of the message. Figure 1 depicts the calculation of the FCS at the source and the sink as well as the message transmitted. If the data at the source (data A) and the sink (data A') are the same, then the ability of CRCs to detect errors (bit flips) on the communication line is *not* influenced. The only consequence of including data A in the CRC calculation is the need to consider the overall length (length of data A and data B) for the achievable Hamming distance (HD). A reason for including some information (data A) into the CRC calculation but not transmitting it might be that this data is already part of a different protocol layer (e.g. the address in TCP/IP), or that agreement on a protocol version needs to be enforced without the expense of communication bandwidth. We call the CRC FCS calculation with additional data that is not sent an *extended CRC check*.

If the data – included in the CRC calculation but not sent – is different at the source and the sink (data A and data A' in figure 1) the ability of the CRC to detect failure in transmitted data is decreased. The decrease is dependent on the number and positioning of bits that are different between data A and data A'. Even a single bit error in data transmission can lead to an undetectable error if more than $HD - 1$ bits differ in the unsent data (see section 3).

Example: Implicit Acknowledgment in TTP/C. The HD of TTP/C is at least 6 [44] for frames that use the CRC to protect 2024 or fewer bits. For certain types of frames (called N-frames) TTP/C calculates the CRC over the data and header (including the version number/schedule identifier and the membership vector), but the version number and the membership vector is not transmitted. TTP/C has a mechanism called implicit acknowledgment [8]. When a sending node experiences a fault that prevents any receiver from receiving its frame, it is placed into a separate group from other transmitters, and the membership vector of nodes in these different groups can differ by up to two bits. Sending nodes then compute their transmitted FCS based on these different membership vectors. Because the Hamming distance of the TTP/C CRC is at least 6, and 2 bits of Hamming distance can be consumed by membership vector differences, this leaves only a Hamming distance budget of 4 bit inversions to detect possible bit errors in transmitted data. This clearly increases the probability of an undetected error.

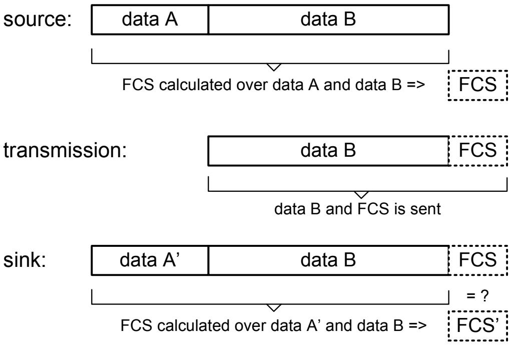


Figure 1. Calculation of an extended CRC check

Example: Group Membership in TTP/C. In TTP/C there is another mechanism that is also based on the membership vector and extended CRC check if N-frames are used. This mechanism is called the clique avoidance mechanism [6]. If a frame is received from some nodes but not from others (a Byzantine fault), two cliques will form. One clique that has received the frame correctly, and one that has not. These cliques will not be able to communicate between each other, because TTP/C requires agreement on the membership vector to be able to receive a frame due to consistency requirements.

In the two TDMA rounds following a Byzantine fault, two node-local counters are increased at each node depending on the CRC check. The accept counter (AC) is increased for a matching FCS, and the reject counter is increased for a failed CRC check. Before a node sends, it checks the relative value of the counters ($AC > FC$) and shuts down when the counters indicate that it is a member of the smaller of the two cliques (this is if $FC \geq AC$). It can take up to two TDMA rounds until all nodes in the smaller clique shut down (assuming no CRC coverage issues). During this period, the membership vector of different cliques and nodes (see [8] for illustrations) differ by nearly up to the number of communicating nodes (called the cluster size).

Since the Hamming distance of TTP/C is 6 and membership bits could be arbitrarily distributed over the 64 bits, this means that for cluster size larger than 6 it is possible that two different membership vectors will alias to an identical CRC computation. If this happens, counters will be inadvertently increased even in the case of fault-free transmissions. For larger clusters this can lead to cliques that survive for a non-analytically determinable time even with no network transmission errors.

As an example, consider a cluster with 32 nodes, which splits into two cliques (with 16 nodes each) due to a Byzantine fault [39]. The probability that a node increases the wrong counter can be calculated as follows: There are 16 minus 6 (=10) possibilities that the wrong counter is increased, since a clique will receive up to 16 frames from the other clique during the first TDMA round following the Byzantine fault and the first 6 are covered by TTP/C's HD. The chance of an undetected CRC error is $2^{-24} = 6 \cdot 10^{-8}$. So the chance of the wrong counter being increased at *any* node is $6 \cdot 10^{-8} \cdot 10 \cdot 32 = 1.92 \cdot 10^{-5}$. This is an effect that is probably not acceptable for ultra-dependable systems, except in situations in which such Byzantine faults can be proven to occur with very low frequency.

It has to be emphasized that the membership-dependent decrease in CRC error detection coverage of TTP/C can only happen with the frame type "N-frames" but not with "X-frames" or "I-frames", because the latter do not "hide" the membership vector by using extended CRC checks.

4.2. Intermediate Communication Stages

The error detection capability of CRCs discussed in section 3.1 might also be compromised by inter-stages in the network. Any active network component that is on the path from the source to the sink of a message is called an *intermediate communication stage* (an *inter-stage*).

If constructed using active components, an inter-stage could encounter a systematic (correlated) error. One example of such a systematic error is a bit flip every several data bits caused by a faulty transceiver circuit, weak driver, partial short, or defective multi-bit buffer register. The complexity of a circuit is not an issue for encountering systematic failure modes, as even simple circuits can exhibit this or similar behavior [24, 31].

The effect of correlated failures on CRC error detection is comparable to bursty and fading channel effects (i.e., a channel with memory). If systems with inter-stages are similar to channels with memory, the often used bound of undetected errors that assumes random independent errors (see section 3.2.1) may not hold.

These potential systematic failure modes of inter-stages are equivalent to source (and destination) failure modes. One key distinction for ultra-dependable systems design lies in the influence that an inter-stage can have on the entire communication system. If the interstage is central and insufficient independent communication paths exist from the data source to the data sink (that is a single or dual path in a single fault-tolerant system), interstages have a significant influence. In contrast, source errors only affect the output of a single device, which can be dealt with known fault tolerance techniques.

Another distinction of inter-stages from source devices is that the potential for Byzantine errors [13] makes it impossible to increase the error-detection coverage of the inter-stage to a level required in ultra-dependable systems by traditional means such as replication of inter-stages (i.e. two devices in a single interstage) and mutual monitoring and controlling of outputs of the devices. For example, one of the two inter-stage devices could be faulty and output a weak signal (a signal with some "1/2 signal values"), where 1/2 values of the signals are interpreted as a one at the nearby correct monitor node but as zero by distant receiver nodes due to different voltage levels of receivers and signal attenuation due to the communication medium. Even triple replication might not work since the required voter of replicated guardian devices could exhibit failure modes with correlation.

In some systems, the inter-stage also recalculates the CRC. In [42], Stone and Partridge find the end-to-end undetected error probability relatively high (up to 1 packet in 16 million packets, which equals an undetected error probability of $62.5 \cdot 10^{-9}$) when evaluating the performance of real networks in which the CRC is recalculated in each inter-stage. The link-level CRC has been found correct for up to 1 packet in 1100 packets when the TCP checksum fails. This high error rate is attributed to hardware and software failures of end-systems and intermediate communication stages. Indeed, Stone and Partridge recommend application-level CRCs (transmitted in the data field and not recalculated by network inter-stages) to help detect transmission errors due to inter-stages.

Even if the CRC is not recalculated, and the signal is "only" reshaped by the inter-stage (such as in [5]), it is questionable whether the error detection probability of CRCs is as strong as for memoryless channels (described in section 3.2.1), where no correlation is assumed.

What has to be assumed for end-to-end undetected error probability for communication systems with inter-stages? This is difficult to tell. An interesting evaluation of end-to-end error detection in computer networks conducted by Lai [30] evaluated the link-level and node-level probabilities of undetected and detected error and found that the end-to-end error detection probabilities are actually dominated by link and node level probabilities, and that the influence of the network medium and inter-stages were minimal. Thus, a safe bound has to assume the device failure rate of the inter-stage is approximately (10^{-6} failures/hour) unless further system-level or component-level mitigation is performed. This value might have to be used even if CRCs are being used with a significantly better undetected error rate.

Any argument that CRCs improve error detection beyond this bound has to hinge on an assertion that CRCs provide a digital signature that is cryptographically secure enough to resist ”attack“ by arbitrary faults. We discuss this point later in section 5.5.

4.2.1. Example: Dual-Bus vs. Dual-Star. In the following, dual-bus and dual-star network topologies are compared purely from a CRC error detection perspective and overall system claims. First, emphasis is placed on a purely probabilistic arguments for CRCs. The comparison is then revisited taking other failure modes into account. Example architectures for dual redundant network configurations are TTP/C [44], FlexRay [16], or AFDX [2]. TTP/C and FlexRay use a 24-bit CRC. Figure 2 depicts a dual-bus and dual-star network.

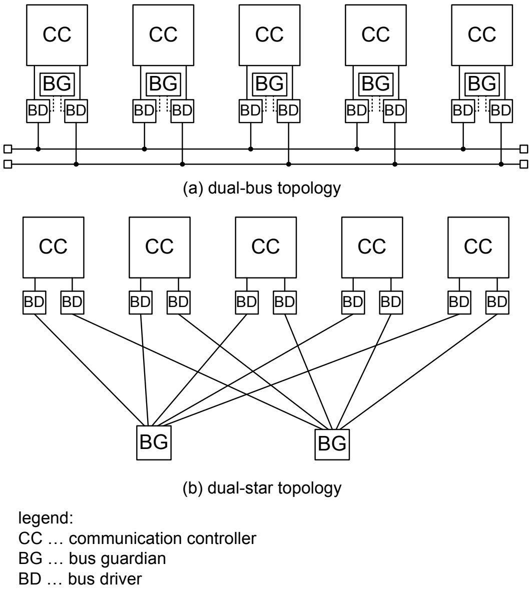


Figure 2. (a) Dual-bus Network and (b) Dual-star Network

In a dual bus topology, CRC error detection is only used for medium-induced errors because the communication path from one communication controller to another does not contain active inter-stages. For medium-induced errors and low bit-error rates, the probability of undetected errors is likely less than $2^{-24} \approx 6 \cdot 10^{-8}$ per message (section 3.1). Fault-tolerance mechanisms on top of the bus can reach at most an integrity value of $6 \cdot 10^{-8}$ per message, if no other error detection mechanism (e.g. physical layer encryption as described in section 5.4) is taken into account. At 10,000 messages per second and a BER of 10^{-6}, this is approximately $2 \cdot 10^{-6}$ failures per hour.

In a dual-star topology with an active inter-stage such a bus guardian with reshaping of the signal[1], the end-to-end error detection of CRCs cannot easily be analyzed, because failures might be uncorrelated. In the worst case, the undetected error probability (and thus integrity of the communication network) is the device failure rate of any active inter-stage, which is around 10^{-6} (section 2). The complexity of the inter-stage is not the primary issue here. The issue is that any device can fail in an arbitrary way, including correlated bit flips. This is a failure mode against which CRCs provide a poor error detection probability.

The recent shift in X-by-Wire protocol emphasis from distributed bus-based architectures towards star-based architectures with active central inter-stages comes along with a potential significant decrease in the achievable dependability. In a bus-based system, the device failure rate of network components can be masked, e.g., by using voting of multiple independent data sources. For a star-based system with active inter-stages, the device failure rate of the inter-stage influences every message. As a consequence of the central element potentially affecting the integrity of every message, voting of independent sources is not effective for nodes connected to any single star hub. If voting can be done at all, it requires voting across multiple star hubs. This transforms the purpose of having multiple physical networks from one of reliability and availability (having a second physical network in case the first one breaks) to one of integrity (having a second physical network to avoid single-point failures that corrupt data in an arbitrary manner).

It has to be stressed that this comparison solely looks at the CRC error detection coverage. A holistic picture should consider the failure probabilities of the different architectures as well. In reality, the likelihood of failures on a bus system may be greater than the likelihood of failures on a star-based system due to several factors, such as: a bus topology has a higher bit error rate due to more connectors that can fail on a per-link basis, additional reflections due to more connectors on the shared common hardware link, increased driver load due to greater bus driver fan out, and common resource failures (spatial proximity failures, babbling nodes). As an example: The probability of a babbling device in a dual-bus configuration with 20 end-components is $2 \cdot 10^{-6} (= 20 \cdot 10^{-7})$ for device failure rates of 10^{-7} causing babbling failures. In a dual star configuration, where the inter-stage detects babbling nodes with sufficiently high coverage (say it detects babbling failures with a coverage of 10^{-3}), the probability of the system being affected by babbling devices is equal to $2 \cdot 10^{-9} (10^{-3} \cdot 2 \cdot 10^{-6})$. Assuming independence of the interstages, the probability of both inter-stages failing at the same time is better than $2 \cdot 10^{-9}$. For such a “babbling” failure mode, dual-star clearly outperforms the dual-bus configuration.

We have to mention again that it is difficult to tell

[1] Note that the passive unconstrained failure mode of central guardians refers to creating messages. While it is possible [31], it is difficult to associate a probability number for creation of messages. Yet, a guardian performs signal reshaping. Any reshaping circuit is active and will influence CRC coverage even with a simple failure mode, if a systematic error is assumed!

what the real device failure rate is that leads to an undetected CRC error of an inter-stage. Yet, some of the authors have seen "unbelievable" failure modes [13], so assuming the worst case is always a good rule to follow in ultra-dependable system design.

5. Design Recommendations

This section explicitly states some guidelines for the use of CRCs in ultra-dependable systems. It also presents design mitigation strategies illustrating how to avoid reliance on CRCs for error detection for sources beyond medium-induced errors. These guidelines and design mitigation strategies are only intended to be used for ultra-dependable systems (i.e., system failure rates of 10^{-9} failures per hour or smaller).

5.1. Use CRCs Only for Medium-Induced Errors

Given that any device in the system can fail in any arbitrary way, including systematic failures that violate the random independent error assumption, CRCs should only be used for error-detection coverage in the network medium.

The end-to-end error detection ability of CRCs for communication paths including active inter-stages is bounded by the device failure rate (i.e. around 10^{-6} failures per hour, section 2). Beyond a certain point, using bigger CRCs doesn't provide additional coverage, because the inter-stage device failure rate dominates.

As explained in section 2, testing is not a means to achieve 10^{-9} assurance levels. These arguments rule out the sole dependence on CRCs for end-to-end error-detection coverage if inter-stages are present, including even inter-stages constructed with passive devices. That is, a system should not solely rely on the use of CRCs to produce signed messages. Building upon this guideline are the recommendations described in the next sections.

5.2. Independent Data Paths for Topologies with Inter-stages

In systems with inter-stages it is necessary to use independent communication paths to achieve sufficient data integrity assurance. The argument is that a checking mechanism that can increase the error detection coverage of the communication medium needs *independent* input (independent from the inter-stages). The following two sections illustrate how this might be achieved:

5.2.1. Voting. An example of the use of voting in order to achieve data integrity is ROBUS (reliable optical bus) from SPIDER [34]. SPIDER may use CRCs to detect errors in the communication medium, but does not depend on the error-detection coverage of CRCs. Figure 3 depicts a single-fault tolerant ROBUS of SPIDER.

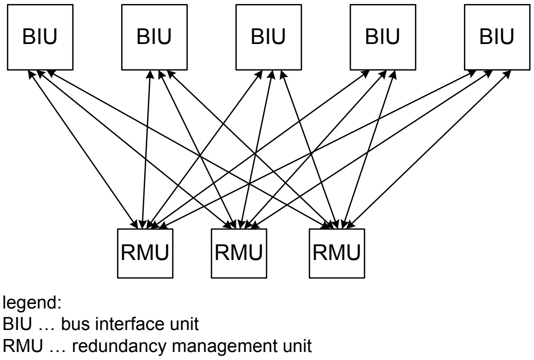


Figure 3. ROBUS of SPIDER: example of voting to ensure data integrity and availability in a single Byzantine failure scenario

The SPIDER architecture contains inter-stages, called redundancy management units (RMUs), between the end nodes, called bus interface units (BIUs). To perform successful majority voting despite F faulty elements (RMU, BIU, or links), SPIDER requires at least $2F + 1$ RMUs and $2F + 1$ BIUs with direct connections between each RMU and BIU. For a single fault-tolerant systems, this means at least 3 RMUs and 3 BIUs. As a consequence of redundancy, majority voting allows bit-for-bit comparison and thus provides data integrity and availability.

5.2.2. Coverage – Braided Ring. Another concept achieving integrity data via independent links and thus no reliance on CRCs to cover inter-stages is the full error-detection coverage approach achieved by an independent link. The coverage approach requires an independent (passive) communication path in case of an inter-stage.

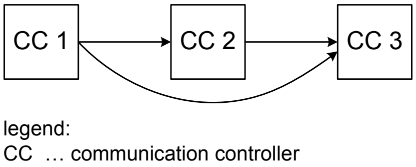
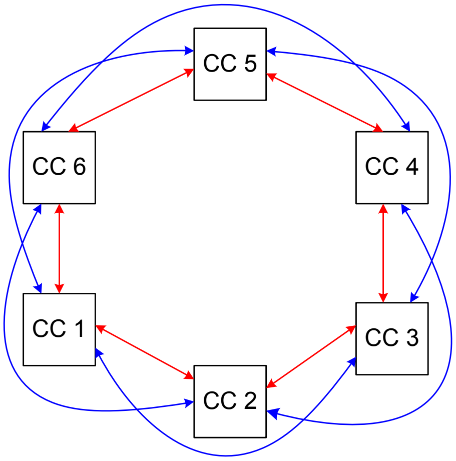


Figure 4. Coverage approach to assurance of data integrity

Figure 4 illustrates the coverage approach for an inter-stage – communication controller 2 (CC 2) – for a data path from CC 1 to CC 3. Data from CC 1 to CC 3 is sent via CC 2 and directly. This allows bit-for-bit comparison at the receiver and ensures data integrity despite of the presence of an inter-stage.

A braided ring uses the full coverage approach to achieve assurance of the error detection coverage for inter-stages [19]. Each node of the ring performs a bit-for-bit comparison of the received data of its inner link and its braid link, which enables high-integrity data transmission without relying on CRC error detection capability.



legend:
CC … communication controller

Figure 5. Conceptual view of a braided ring using coverage approach building blocks

The braided ring uses the reverse transmission direction to achieve communication availability. The ring can use CRCs to detect medium-induced errors without influence of systematic failures of inter-stages. The connectivity in the ring allows neighboring devices to work as fully independent guardians without the expense of additional devices. Figure 5 shows a conceptual view of the ring. Cabling of the braid can be routed via the inter-stage, which saves wires and cabling effort. The braided ring achieves equivalent integrity levels as SAFEbus^{TM} [21].

5.3. Careful Analysis of Extended CRC Checks

Any change in the data of an extended CRC check (that is, any change in the untransmitted data) must be detectable by CRCs. That is, the length of any sequence of bits in the untransmitted data that can differ between the source and destination must be limited to the length of the CRC, and the total number of bits that can differ between the source and destination must be smaller than the Hamming distance.

The interactions of extended CRC checks and medium-level coverage should be studied in detail. Once the data for the extended CRC check or the CRC initialization value at the sender and receiver differ, the error-detection coverage of CRCs on the physical layer is decreased, because any additional bit errors on the physical layer may exceed the error detection capability of the CRC.

The practice of using different initialization values (also called seed values) to provide separate virtual communication channels has similar issues to an extended CRC check. From an error detection perspective, using different initialization values for computing CRCs is identical to having untransmitted data equal to the CRC size. Both techniques reduce the effective Hamming distance available by overloading CRC operation with both data error detection within a virtual channel and prevention of spillover between virtual channels in the event of bit errors.

Extended CRC checks and/or different initialization methods should only be used where the consequences are studied in detail. An example for using the different initialization values in the CRC calculation in an acceptable way can be found in TTP/C, which initializes the CRC block differently for different channels. The initialization values are a function of the schedule identifier of the TDMA access table (Message Descriptor List) in TTP/C. This allows detection of crossed-out channels and incompatible schedules. The use of CRCs to prevent incompatible schedule tables seems to be adequate and will be detected with a high probability.

5.4. Physical Layer Encryption

Instead of using CRC initialization values or extended CRC checks, physical layer "encryption" can be a design alternative (or additional mechanism) to achieve a good error-detection coverage. Crossed channels (two channels mistakenly connected to the wrong connectors on a physical device) are one type of error that can be detected with physical layer encryption. Two channels are considered crossed out if the signal on one is the inverted signal on the other. Since start of message signals normally use out-of-band encodings, channels connected backward will send invalid (inverted polarity) start of message signals, and thus be detected independently of the CRC.

For example, in TTP/C this means that if one channel uses a specific physical medium level decoding then the other channel could use the inversion of the first without affecting the error detection coverage of CRCs. Using different bit encoding schemes on two channels (e.g., Manchester vs. NRZ encoding) is another common approach. Please note, however, that certain physical layer decoding techniques, such as *bit stuffing* can influence the error detection coverage of CRCs, as illustrated for CAN in [43].

5.5. Cryptographic Techniques

There have been many suggestions in the literature to provide message integrity and authentication by using cryptographic integrity methods similar to CRCs. These methods append either a message authentication code or a digital signature to a message. The distinction between these methods is that the latter uses public-key cryptography. These suggestions always include the required assumption that the appended data is "unforgeable". Alternatively, one could use a CRC and append the message sender's ID to each message, either explicitly or implicitly (e.g. use the ID as the CRC seed). The CRC method typically is 10 to 1000 times less costly in system resources (CPU time, memory, hardware area) than the cryptographic methods.

Cryptographic integrity methods do not provide fault detection benefits to ultra-dependability that are commensurate with their additional costs. In fact, in one respect, the cryptographic methods are weaker than a CRC method. This is because the cryptographic methods cannot guarantee a Hamming distance greater than one. If this were not the case, the method would be cryptographically weak. An interesting issue is the degree to which the "unforgeable" assumptions hold. One fallacy is that cryptographic methods with a secret key are harder to forge. In reality, it makes no difference to natural failure mechanisms whether the key is kept secret or is published in all the world's newspapers. For equal size ID/key, the probability of success for "brute force attacks" is the same for CRCs as it is for any cryptographic method. The remaining assumption question is: Is it more probable that a device failure will emulate a simple CRC mechanism than a more complex cryptographic mechanism? The question is moot. The probability of either is so small that their difference is unquantifiably infinitesimal. That is, one cannot gather enough statistical evidence to differentiate between the probabilities. For this reason, many certification agencies say that one cannot differentiate among failure behaviors for devices with very small failure rates. That is, one must assume that when such a device does fail, it fails such that it does exactly what you don't want to do. An example is a failure that results in a device state and/or structure change that then can forge messages. Thus, there is no distinction between the failure probabilities for CRC versus cryptographic mechanisms.

6. Conclusions

Cyclic Redundancy Codes (CRCs) are a proven approach to detecting errors for random, independent bit inversions of the type often seen on passive communication media. They can help bring communication system undetected error rates down to the level of hardware component failure rates.

However, the use of CRCs as a mechanism to provide ultra-dependable system operation (10^{-9} failures/hour) is questionable in many cases. The main problem is that network inter-stages can exhibit arbitrary faults, accidentally forging valid CRC check sequences. These faults can dominate system dependability issues, resulting in undetected failures at the 10^{-6} component failure rate.

Getting beyond the component failure rate limit requires architectural methods. Moreover, cryptographic methods cannot be argued to be strong for single arbitrary component failures.

An additional consideration in the use of CRCs is that the inclusion of "hidden" state in CRC computations can compromise error detection properties. This includes both extended CRCs, in which hidden state is included in a CRC computation, as well as diverse initialization (seed) values intended to provide logically separate communication channels over a single network. Careful consideration of Hamming distance capability as well as potential differences in hidden state computations is required to avoid compromising error detection effectiveness if a data transmission error happens to combine with a hidden state difference to produce an undetected error.

7. Acknowledgements

This work is supported in part by the General Motors Collaborative Research Laboratory at Carnegie Mellon University, the American Association of University Women and Zonta Int. fellowship programs, a National Science Foundation Graduate Research Fellowship, and the Federal Aviation Administration, Aircraft Certification Service (AIR-1), 800 Independence Ave., S. W., Washington, DC 20591.

Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation or the Federal Aviation Administration.

References

[1] K. Abdel-Ghaffer, "A Lower Bound on the Undetected Error Probability and Strictly Optimal Codes", *Information Theory, IEEE Trans. on*, Sept. 1997, pp. 1489–1502.

[2] Airlines Electronic Engineering Committee (AEEC), "Part 7: Avionics Full Duplex Switched Ethernet (AFDX) Network (Draft 3)", *Project Paper 664: Aircraft Data Network*, Aeronautical Radio, Inc., Annapolis, MD, USA, Sept. 2004.

[3] Allen-Bradley, "Tech Talk: A Giant Leap in Noise Immunity: Unshielded is NOT Unhealthy", *Sensors Today*, Rockwell Int. Corp., June 1999, pp. 8–13.

[4] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic Concepts and Taxonomy of Dependable and Secure Computing", *Dependable and Secure Computing, IEEE Trans. on*, IEEE Press, Jan. 2004, pp. 11–33.

[5] G. Bauer, H. Kopetz, and W. Steiner, "The Central Guardian Approach to Enforce Fault Isolation in a Time-Triggered System", *Autonomous Decentralized Systems, Proc. of the* 6^{th} *Intl. Symp. on*, Pisa, Italy, Apr. 2003, pp. 37–44.

[6] G. Bauer and M. Paulitsch, "An Investigation of Membership and Clique Avoidance in TTP/C", *Reliable Distributed Systems, Proc. of the* 19^{th} *IEEE Symp. on*, IEEE Press, Nuremberg, Germany, Oct. 2000, pp. 118–124.

[7] V. Blinovsky, "New Estimation of the Probability of Undetected Error", *Information Theory, Proc. of Int. Symp. on*, IEEE Press, 17–22 Sept 1995, p. 57.

[8] A. Bouajjani and A. Merceron, "Parametric Verification of a Group Membership Algorithm", *Formal Techniques in Real-Time and Fault-Tolerant Systems, Int. Symp. on*, LNCS Vol. 2469, Springer-Verlag Heidelberg, Sept 2002, pp. 311–330.

[9] R. Butler and G. Finelli, "The Infeasibility of Quantifying the Reliability of Life-Critical Real-Time Software", *Software Engineering, IEEE Trans. on*, Jan. 1993, pp. 3–12.

[10] G. Castagnoli, S. Brauer, and M. Herrmann, "Optimization of Cyclic Redundancy-Check Codes with 24 and 32 Parity Bits", *Communic., IEEE Trans. on*, June 1993, pp. 883–892.

[11] E. Chan, Q. Le, and M. Beranek, “High Performance, Low-Cost Chip-on-Board (COB) FDDI Transmitter and Receiver for Avionics Applications”, *Electronic Components and Technology Conf., Proc.*, IEEE, May 1998, pp. 410–17.

[12] C. Constantinescu, “Trends and Challenges in VLSI Circuit Reliability”, *IEEE Micro*, July–Aug. 2003, pp. 14–19.

[13] K. Driscoll, B. Hall, M. Paulitsch, P. Zumsteg, and H. Sivencrona, “The Real Byzantine Generals”, *IEEE Aerospace and Electr. Systems Soc.*, Salt Lake City, UT, USA, Oct. 2004.

[14] K. Driscoll, B. Hall, K. Sivencrona, and P. Zumsteg., “Byzantine Fault Tolerance, from Theory to Reality”, *Comp. Safety, Reliability and Security (SAFECOMP 2003), Conf.*, LNCS 2788 Springer-Verlag, Sept. 2003, pp. 235–248.

[15] Federal Aviation Administration, U.S.Dept.of Transp., “Part 25 Airworthiness Standards: Transport Category Airplanes. Section 25.1309”, *Code of Fed.Reg. Title 14: Aeronautics and Space*, U.S. Gov. Print. Off., 2004, pp. 456–457.

[16] FlexRay Cons., *FlexRay Communications System: Protocol Spec. Ver.2.0*, `http://www.flexray-group.com/` (accessed on 2004-12-03), June 2004.

[17] F. Fu, T. Kløve, and V. Wei, “On the Undetected Error Probability for Binary Codes”, *Information Theory, IEEE Trans. on*, IEEE Press, Feb. 2003, pp. 382–390.

[18] T. Fujiwara, T. Kasami, and S.-P. Feng, “On the Monotonic Property of the Probability of Undetected Error for a Shortened Code”, *Information Theory, IEEE Trans. on*, IEEE Press, Sept. 1991, pp. 1409–1411.

[19] B. Hall, K. Driscoll, M. Paulitsch, and S. Dajani-Brown “Ringing Out Fault Tolerance: A New Ring Network for Superior Low-Cost Dependability”. In *Dependable Systems and Networks, Conf.*, Yokohama, Japan, June–July 2005.

[20] G. Heiner and T. Thurner, “Time-Triggered Architecture for Safety-Related Distributed Real-Time Systems in Transportation Systems”, *Fault-Tolerant Computing Symp, Proc. of*, IEEE, Munich, Germany, June 1998, pp. 402–407.

[21] K. Hoyme and K. Driscoll, “SAFEbus$^{\text{TM}}$”, *IEEE Aerospace and Electronics Systems Mag.*, IEEE, Mar. 1993, pp. 34–39.

[22] International Electronic Commission (IEC), *IEC 61508: Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems.*, IEC, `http://www.iec.ch/` (accessed on 03 Dec 2004), 1998–2004.

[23] International Electrotechnical Commission, *IEC 61375-7, Annex B: Guidelines for Conformance Testing*, IEC, `http://www.iec.ch/` (accessed on 03 Dec 2004), May 1998.

[24] H. Konuk and F. Ferguson, “Oscillation and Sequential Behavior Caused by Interconnect Opens in Digital CMOS Circuits”, *Test Conference, Proc. of the IEEE Int.*, IEEE Comp. Soc. Press, Nov. 1997, p. 597.

[25] P. Koopman, “32-bit cyclic redundancy codes for Internet applications”, *Dependable Systems and Networks, Proc. Int. Conf. on*, IEEE, Wash., DC, USA, June 2002, pp. 459–468.

[26] P. Koopman and T. Chakravarty, “Analysis of the Train Communication Network Protocol Error Detection Capabilities”, Technical Report, Carnegie Mellon University, Pittsburgh, PA, USA, Feb. 2001.

[27] P. Koopman and T. Chakravarty, “Cyclic Redundancy Code CRC Polynomial Selection For Embedded Networks”, *Dependable Systems and Networks, Int. Conf. on*, IEEE, Florence, Italy, 28 June – 1 July 2004, pp. 131–140.

[28] P. Koopman, E. Tran, and G. Hendrey, “Toward Middleware Fault-Injection for Automotive Networks”, *Fast Abstract. Fault-Tolerant Computing Systems, 28th Int. Symp. on*, IEEE Comp. Soc. Press, Munich, Germany, June 1998.

[29] A. Kuznetsov, F. Swarts, A. Vinck, and H. Ferreira, “On the Undetected Error Probability of Linear Block Codes on Channels with Memory”, *Information Theory, IEEE Trans. on*, IEEE Press, Jan. 1996, pp. 303–309.

[30] W. Lai, “End-to-End Probability of Undetected Error in Computer Networks”, *Communications, IEEE Trans. on*, IEEE Press, Mar. 1986, pp. 291–293.

[31] D. B. Lavo, T. Larrabee, and B. Chess, “Beyond the Byzantine Generals: Unexpected Behaviour and Bridging Fault Diagnosis”, *Test Conference, Proc. of Int.*, IEEE, Washington, DC, USA, Oct. 1996, pp. 611–619.

[32] S. Leung-Yan-Cheong and M. Hellman, “Concerning a Bound on Undetected Error Probability”, *Information Theory, IEEE Trans. on*, IEEE Press, Mar. 1976, pp. 235–237.

[33] S. Lin and D. Costello, *Error Control Coding: Fundamentals and Applications*, 2nd edition, Prentice Hall, Englewood Cliffs, NJ, USA, 2002.

[34] P. Miner, M. Malekpour, and W. Torres, “A conceptual design for a Reliable Optical Bus (ROBUS)”, *Proc. of the* 21^{st} *Digital Avionics Systems Conference*, IEEE Press, Hampton, VA, USA, 2002, pp. 13D3–1 – 13D3–11.

[35] W. Peterson and E. Weldon, *Error-Correcting Codes*, 2nd edition, MIT Press, Cambridge, MA, USA, 1972.

[36] D. Powell et al. *A Generic Fault-Tolerant Architecture for Real-Time Dependable Systems*, p. 45ff. Kluwer Academic Publishers, 2003.

[37] RTCA, Inc., *RTCA/DO-178b: Software Considerations in Airborne Systems and Equipment Certification*, RTCA, Washington, DC, USA, Dec. 1992.

[38] RTCA, Inc., *RTCA/DO-254: Design Assurance Guidance For Airborne Electronic Hardware*, RTCA, Washington, DC, USA, Apr. 2000.

[39] H. Sivencrona, P. Johannessen, and J. Torin “Protocol Membership Agreement in Distributed Communication System – A Question of Brittleness”. In *SAE World Congress*, number 2003-01-0108, Detroit, MI, USA, Mar. 2003. SAE.

[40] R. Stephens, “Analyzing Jitter at High Data Rates”, *IEEE Optical Communications*, Feb. 2004, pp. S6–S10.

[41] J. Stone, M. Greenwald, C. Partridge, and J. Hughes, “Performance of Checksums and CRC’s Over Real Data”, *Networking, IEEE/ACM Trans.*, ACM, Oct. 1998, pp. 529–543.

[42] J. Stone and C. Partridge, “When the CRC and TCP checksum disagree”, *Applications, Technologies, Architectures, and Protocols for Computer Communication, Proc. of the Conf. on*, ACM, Stockholm, Sweden, 2000, pp. 309–319.

[43] E. Tran “Multi-Bit Error Vulnerabilities in the Controller Area Network”, Master’s thesis, Carnegie Mellon University, Pittsburg, PA, USA, 1999.

[44] TTTech Computertechnik GmbH, *Time-Triggered Protocol TTP/C*, TTTech Comp. GmbH, Vienna, Austria, 2004.

[45] U.S. Dept. of Defense, *MIL-HDBK-217F: Reliability Prediction of Electronic Equipment*, DAPS, Philadelphia, PA, USA, Feb. 1995.