Communication Integrity in Networks for Critical Control Systems

A. Youssef, Y. Crouzet, A. de Bonneval, J. Arlat

LAAS-CNRS, University of Toulouse
7, Avenue du Colonel Roche 31077 Toulouse Cedex 04 – France
{youssef,crouzet,debonneval,arlat}@laas.fr

J.-J. Aubert, P. Brot

Airbus France
316, Route de Bayonne 31060 Toulouse – France
{jean-jacques.aubert, pascal.brot}@airbus.com

Abstract

The paper proposes a solution to the problem posed by the inclusion of complex interstage nodes into communication networks. Thus nodes are prone to exhibit failure modes that may lead to repetitive errors that breach the usual set of assumptions considered for medium-level protection mechanisms. The specific class of application targeted (e.g., servomechanisms controlling flight control surfaces) is characterized by: i) slow dynamics of the controlled processes does not impose a high integrity level for each transmitted message, but rather for a set of successive messages, ii) the reference values assigned during the mission (e.g., cruise phase) are expected to be maintained identical for several cycles, and thus should the protection fail for one message, then it will be the same for subsequent messages. To cope with this, the proposed integrity protection scheme features distinct error coding functions, thus providing complementary detection capabilities to consecutive messages.

Keywords: safety-critical systems, digital communications, interstage nodes, integrity, error detecting codes.

1 Introduction

Communication integrity is an important requirement for many critical control application domains for which digital networks are increasingly being used. This trend is mainly driven by the deployment of "smarter" field instruments that feature built-in microprocessors and are capable of running more complex control algorithms.

Future commercial aircrafts provide good examples of such critical distributed embedded systems for which an increasing number of sensors and actuators are being deployed: plans exist where the range of tens of control surfaces would be rapidly augmented to the hundred-range or more. The goal is in particular to ensure a more efficient management by the Flight Control System (FCS). The FCS is characterized by very stringent dependability requirements; a commonly quoted figure for the failure rate is 10^{-9} failures/hour.

In such a context, dependability focuses on the provision of high integrity (i.e., the "absence of improper state alterations" [1]). At the level of communication systems, integrity refers to the messages conveyed on the network that links the considered devices. This means that communication failures should — at least — be detected with sufficient high probability.

Communication integrity is often obtained by using Cyclic Redundancy Codes (CRC) [2] that offer a suitable approach to detect data corruption across communication networks [3]. However, the use of CRC alone is seldom sufficient to achieve the integrity levels suitable in such a context, as was already exemplified in [4]. In particular, communication networks are not only featuring passive network interstages, which is prone to breach the error models that are assumed usually: i.e., random independent symbol errors. The classical approach, that relies on increasing the number of check bits, has a strong impact on the "yield" of the resulting code, due to the small size of the messages for the avionics application considered here (≈ 100 bits).

Thanks to the slow dynamics that govern current FCS, it is not necessary to aim at a high integrity level for each transmitted message, but rather for a set of messages. Accordingly, we are proposing an original protection scheme for achieving high communication integrity at the application level: distinct error detecting codes featuring complementary error detection capabilities are applied to consecutive messages.

Section 2 identifies the type of control systems being targeted, the error assumptions that apply to the associated communication channels, and the challenges posed to the classical usage of error detecting codes (e.g., CRC) in this context. Section 3 discusses relevant related work and sets up the guidelines for the proposed error checking scheme. Section 4 sets the principle of the proposed scheme and describes how distinct error checking functions can be used to exploit the applicable assumptions, while still achieving high communication integrity. Section 5 illustrates how this scheme can be implemented by using CRC. Section 6 presents some results of the validation studies carried out to support the proposed scheme. Finally, Section 7 provides our concluding remarks.

2 Problem Statement

For critical control systems, the rate of occurrence of the failures, considered as undesired event, has to be maintained below a specific threshold. For the category of ultra-dependable systems we are dealing with, the FCS for commercial aircrafts, this threshold is set to 10^{-9}/h, which corresponds to the highest level identified in relevant normative documents [5, 6]. A similar value is being considered by the automotive industry for the "X-by-wire" systems [7, 8], which is mostly based on the cross-domain international standard IEC 61508 [9], even if domain-specific standards are emerging now.

This section describes first the main features of the class of control system we are considering. We then focus on the communication network that is one core component in the digital architecture that supports the control system. We also motivate the approach proposed via a brief analysis of the risks that are induced by the limitations of low-level protections to address the challenges posed.

Proceedings of the Sixth European Dependable Computing Conference (EDCC'06)
0-7695-2648-9/06 $20.00 © 2006 IEEE

2.1 The class of control system

We introduce the several dimensions that specify the requirements for the type of control system addressed.

2.1.1 Slow/fast dynamics & state/event commands

A process is said to exhibit *slow dynamics* when the lapse time for a significant change to affect the parameters being controlled is large with respect to the duration of the control cycle. Thus, the controlled process is potentially insensitive to the application of a certain number[1] of erroneous reference values, before an undesired event can be reached. In particular, this is the case for the hydraulic actuators that govern the flight control surfaces on commercial airplanes. These devices are dimensioned so that movements of the related surfaces are slow enough. It is worth pointing out that in the case of systems featuring *fast dynamics*, the processing of a single erroneous command might well lead to the undesired event.

A second important feature depends on whether the commands convey an absolute or a relative semantics with respect to a reference value: they are usually referred to as state- and event-controlled systems, respectively. More precisely, for a state-controlled system, a command (called *state-command*) is defined as an absolute reference value of a parameter (e.g., move to the 5 m location or rotate to reach the 5° angular position). For an event-controlled system, a command (*event-command*) is defined as a relative reference — most often with respect to the current value of the command (e.g., move of 5 m, rotate by 5°).

When reference values are to be maintained in the control loop, they are necessarily implemented as state-commands in order to prevent from the related effects to be cumulated. Conversely, an event-command should not be lost or sent twice. These features have an impact on the way protection approaches can be devised: i) in the latter case, it is necessary that protections be provided against each error manifestation, ii) in the first case, one could simply aim at providing protection against a specific number of error occurrences as several commands (including erroneous ones) can be accounted for by the actuators without provoking an undesired event.

Dealing with slow dynamics and state commands altogether has a significant influence on the properties required to support communication integrity: in particular, message loss, duplication, ordering have not to be worried about.

2.1.2 Undesired event and recovery strategy

An *undesired event* (UE) corresponds to an event prone to lead to the failure of the application. A typical example, in the context of avionics, is the runaway of some of the critical flight control surfaces that may result from the steady application of erroneous commands on the corresponding servomechanisms.

As already mentioned, the FCS should be designed so that such a risk is lower than 10^{-9} per hour of flight. In particular, it is considered that for servomechanisms governing critical control surfaces (elevator, rudder) with a maximum speed movement of 50°/s, a runaway may be observed already when the discrepancy with respect to the nominal reference value reaches about 5°, thus when erroneous references are applied for 100 ms. This constitutes actually a very stringent constraint.

[1] Up to a given threshold, depending on the application.

For a flight control surface, whose position is updated every 10 ms, the **UE** corresponds to the acquisition of 10 consecutive incorrect (and not signaled as such) commands that are thus processed as correct ones.

It is worth noting that the actual occurrence of an undesired event also depends on the *recovery strategy* that is undertaken upon detection of erroneous commands. In the case of a FCS, recovery actions are planned at the application level with the aim of mitigating several concerns:

- ensure the correct updating of the reference value to the servomechanism,
- refrain from deciding too quickly about the failure of the communication system,
- do not impair the required safety level.

The frequency with which a recovery action is initiated is related to the error detection rate; the latter being linked in turn to the error occurrence rate. In a communication network, most frequent errors are related to the noise on the channel. Thus, there is a risk that the communication channel be declared as failed too frequently, should this decision be taken each time an erroneous message is identified. Moreover, due to the flexibility allowed by the slow dynamics of the controlled process, no major risk would result in delaying (to some extent) the updating of the reference value.

Thus, in practice, the strategy that is implemented is as follows. Upon detection of an error, the current reference value is maintained. This is fully supported by the slow dynamics that characterize the controlled process (see Section 2.1.1). Such an approach is custom in control systems. Furthermore, a failure is declared and the suitable recovery action launched, only after several related errors have been reported. Such a form of "filtering" is also usual practice. Among the classes of recovery strategies that can be considered as suitable alternatives to the launch of the recovery as soon as an error is signaled, we have especially considered the two following ones:

- **RS1(r):** Maintain the current reference value and launch the recovery after r *consecutive* processing cycles for which an error has been signaled.
- **RS2 (r,b):** Maintain the current reference value and launch the recovery after r processing cycles for which an error has been signaled out of a set of b successive cycles.

However, it is not possible to wait too long before identifying a communication link as failed, because this may then impair the safety requirements. Figure 1 shows a possible scenario[2] that spans ten 10 ms-cycles:

> *The erroneous command is not detected (ND) on the 1st cycle, then detected (D) on the 2nd and 3rd ones, but not on the 4th, etc.*

Let us assume also that the recovery parameters are as follows: r = 3 and b = 10. It is worth pointing out that the rationale for selecting b = 10 is directly related to the definition of the UE for the FCS.

Such a scenario would be critical in the case of RS1(3); in fact, it corresponds to the worst case with the value selected for parameter r: the conditions for initiating a recovery action are never met and thus an erroneous reference command would be maintained during 10 cycles (100 ms),

[2] This scenario might seem rather improbable; actually, this is not the case, in the light of the error checking scheme proposed in Section 4.2.

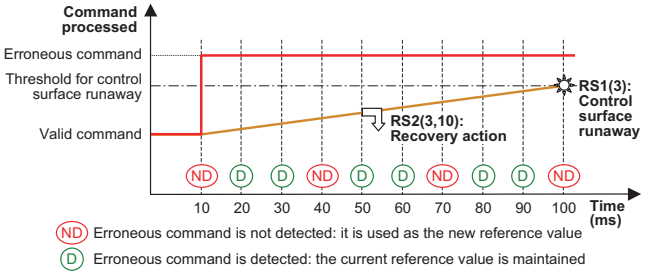

Figure 1: Link between recovery and UE (r = 3, b = 10)

then leading to the runaway of the control surface. However, when RS2 is instantiated with $r = 3$ and $b = 10$, then the conditions for launching the recovery are met at time = 50 ms and thus the runaway can be avoided.

In the case of a digitally controlled servomechanism, the reference values are conveyed by means of messages. Accordingly, it can be considered that the servomechanism receives a message that contains the value of the reference for each cycle. When the message received is corrupted, and the corruption is not detected, an erroneous reference is used; if the corruption is detected then the reference is not applied and the reference value of the servomechanism is not updated. To avoid the UE (control surface runaway), that would result from an erroneous reference value to be maintained for 100 ms (ten 10 ms cycles), it is necessary that the rate of undetected erroneous messages be lower than a given threshold. Here, it is assumed that the UE may result from the fact that $r = 3$ erroneous messages out of a set of $b = 10$ messages remains undetected. Thus, the communication system has to maintain the rate for such a risk of undetected errors lower than 10^{-9}/h.

2.2 Risk analysis

In the context of the type of control systems briefly described in the previous section, the focus of the work reported here concerns the underlying communication networks and in particular in the light of the evolution of such control systems. For communication networks where reference values are conveyed via messages, the concepts of state-control or event-control are usually referred to as state messages or event messages [10]. The usage of state messages is often preferred in the context of critical systems as the control is less sensitive to message loss or incorrect ordering of messages. In what follows, we will essentially refer to state messages.

Specific concerns arise when a large number of devices are to be controlled in a flexible way, which is the case for future developments being considered in avionics, e.g., the deployment of a large number of control surfaces on the wing of an aircraft: i) all devices cannot be connected to the same bus, ii) such future developments include the opportunity to apply distinct commands on different actuators.

In practice, this means that intermediate communication stages (interstage nodes), which are more complex than simple repeaters, are to be included between the control nodes (main control computers) and the controlled devices (computerized sensors and actuators).

Figure 2-a sketches the basic architecture considered for the control system and identifies the core part: the digital communication links that are connecting the main control nodes to the controlled nodes. Nowadays, many versions of

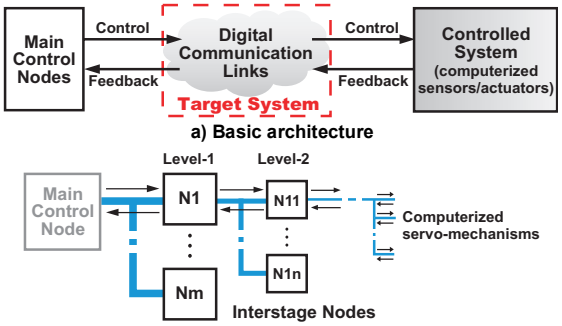

Figure 2: Communication architecture for the control system

the fieldbus technology exist that implement such a concept according to the IEC 61508 standard [9]. Benefits of using the fieldbus technology in control systems are manifold and extend from low costs, flexibility, improved performance and enhanced diagnosis capabilities [11].

Figure 2-b illustrates a possible hierarchical architecture for the communication system that would be suitable for the kind of application we are considering.

Due to the presence of interstage nodes, information coding (e.g., CRC) at the physical layer is no longer sufficient to reach the integrity assigned for ultra-dependable control systems. This problem was already identified in [4].

The analyses we have conducted confirm these results and can be summarized as follows [12]:

1) Noise-induced corruptions (*independent random errors*) are covered with a sufficient margin (several orders of magnitude).
2) Wiring defects (*independent multiple errors*) lead to a risk for undetected errors for a single message that is just slightly acceptable. Two reasons explain why, in our case, this risk remains way below the target threshold: i) the risk analysis relates to a set of messages and ii) these corruptions do not provoke repetitive errors.
3) Corruptions resulting from faults (e.g., stuck at faults) affecting the interstage nodes are prone to lead to recurrent error patterns (*repetitive multiple errors*) for which the risk of undetected error can no longer be maintained under the prescribed threshold. For stuck-at faults affecting the buffers handling messages, the protection offered by CRC could be tuned to match the assigned level, at the expense of increasing the number of check bits. For the faults affecting the address control of the buffers, this is no longer possible: the risk associated to the corruption of a set of messages remaining undetected is of the same order of magnitude as the risk associated to a single message. Indeed, should the initial error be undetected, then systematic (correlated) errors are likely to occur[3]. Thus, the basic assumption of random independent errors usually assessed to bound the risk of undetected errors by using CRC no longer holds.

Assuming a failure rate of 10^{-5}/h for an interstage node, and considering a 1% ratio for the devices prone to lead to repetitive errors, would lead to an error rate of 10^{-7}/h, which definitely requires that alternate protection means be implemented to meet to the assigned target.

[3] Another factor for recurrent errors is that for a large part of flight time (e.g., cruise) the reference value remains unchanged.

3 Related Work

Solutions devised to cope with the inefficacy of medium-level protection mechanisms (e.g., CRC) to handle the errors induced by interstage nodes evidenced in the previous section encompass two main categories of complementary and non exclusive techniques: i) application of end-to-end protection at the application level and ii) use of redundant communication topologies. In addition, it is several forms of diversification of data handling and/or coding can be used to minimize the risk of common mode failures.

3.1 Application-level end-to-end protection

Such an approach assumes an abstraction of lower layers and allows implementing protections that are independent from the (potential) protection mechanisms attached to the lower layers. Some typical examples are given hereafter:

- Time redundant execution of application tasks and end-to-end CRC calculation at sender node support the *High Error Detection Coverage* (HEDC) proposed for systems based on the TTA and TTP concepts [13].
- The *Keyed CRC* mechanism of the GUARDS system relies on the provision of unforgeable signatures; each node uses a private "key" for communicating with each of the other nodes [14].
- The *Safety Layer* supported by the *Foundation Fieldbus* features an end-to-end CRC protection and a two-way protocol to handle safety frames [11].

3.2 Redundant communication topologies

Examples of solutions based on redundant communication topologies are as follows:

- The dual self-checking busses are instrumental in supporting the high communication integrity procured by the *SAFEbus* architecture [15].
- The *ROBUS* broadcast bus of the SPIDER platform relies on point-to-point communication links and voting to ensure reliable communication among all pairs of correct processing nodes in the system [16].
- Elaborating upon a series of designs incorporating various forms of redundant ring topologies, (e.g., see [17]), the *braided ring* topology presented in [18] achieves the same level of integrity as the SAFEbus at a lower cost, compatible with the automotive industry.

3.3 Use of diversification

Two main non-exclusive dimensions can be identified for applying diversification in order to cope with the risk of recurring errors induced by interstage nodes:

- usage of alternative formats to describe data,
- usage of alternative checking functions.

A typical example of the first dimension is when the data are transmitted three times, via distinct channels: i) original bits, ii) inverted bits, and iii) permutated bits [19]. Also, approaches have been devised that include schemes for carrying out voting on diversified data [20]. The *Tandem-CRC* approach [21], where two distinct CRC functions are applied to a message to be transmitted to obtain high coding gains provides a good example of the second dimension. The *Turbo-Codes* [22], that combine data interleaving and concatenation of distinct systematic convolutional codes form a typical example of an approach spanning the two dimensions.

3.4 Design guidelines

Application-level end-to-end approaches allow extending the coverage of errors beyond basic data. This is a very relevant feature for the kind of messages we are considering that include data, address and timing information. However, achieving the level of integrity required for the considered application would require a very large CRC code size (at least 32 check bits) to decrease the probability of undetected error to a suitable level. Actually, this would correspond to a costly solution in terms of "yield" (referring to the size of the functional message: about 100 bits) and of the communication latency. Most importantly, this would not adequately address the issue of repetitive errors in our context: if the protection is not efficient for a message, it is likely that it will be the same for subsequent messages.

All approaches using *redundant topologies* (using or not diversification) for achieving high-level of communication integrity in spite of faults affecting interstage nodes heavily rely on spatial redundancies. This is in opposition with the strong industrial constraint imposed in the context of the system being considered for the deployment of digital networks governing the flight control surfaces, that is to avoid as much as possible the replication of components. In particular, this led us to discard approaches based upon the provision of fail-silent nodes [10] or of nodes tolerating transient faults [23].

As already explained, another important feature of the aircraft application context being considered is that it is not necessary to ensure that each message delivered is correct. In summary, the two main design assumptions that singularize the type of potential solution that is being investigated are as follows:

- **A1**: it is not necessary to reach a high probability of error detection for each message, but rather to a series of successive messages;
- **A2**: the integrity issue posed by repetitive errors is to be addressed while minimizing the use of redundancy.

Besides this might not be true for event-controlled systems, we advocate that A1 is generally valid for state-controlled systems featuring slow dynamics (e.g., temperature control or fluid flow control). Also, maintaining a constant input reference value (e.g., during cruise phase) may favor the occurrence of repetitive errors in the case of faults affecting the interstage nodes. But, similar error types may be provoked even if only some field of the message frame is maintained unchanged: e.g., errors induced by permanent faults (stuck-at faults in memory buffers) in interstage nodes affecting the unchanged part of the frame.

The rationale for the second assumption is also related to the fact that the desired goal is only to support error detection as error recovery (fault tolerance) can be achieved by separate means.

These guidelines open new alternative options for developing suitable solutions, in particular based on the principle of diversification applied to the error checking functions, as will be described in the next section.

4 The Proposed Solution

The solution we propose relies on the implementation of end-to-end protection mechanisms at the application level. After presenting the basic principles of this scheme, we then analyze the impact of the recovery strategy.

4.1 Basic principles

To fulfill the requirements previously stated, the basic idea is not to aim at providing a high probability of detecting errors for each message, but rather for a set of messages.

We propose that sender and receiver modify the checking function for each message, according to a common policy known by them. More concretely, the sending node can use m error checking functions $F_1, F_2,..., F_m$, each generating the same number of control bits. We term this technique as the "Multiple Error Checking Function" scheme (MECF). The classical "Single Error Checking Function" scheme is denoted SECF.

Of course, the sender and the receiver should use the checking functions consistently both for coding and decoding. In the case of synchronous behavior, this can be achieved by using either the cycle index or the local clock for deciding which function to use. In the case of asynchronous nodes, it is necessary to include explicitly the identifier of the function as part of the message.

We focus here on the analysis on the risk of repetitive errors caused by faults affecting interstage nodes. Let us denote M_i the messages in the set (i = 1, 2,..., b). We assume that M_i corresponds to an erroneous message and that the same error affects all subsequent messages in the set. We denote:

- p_{XNDi} the basic probability of no detection of an erroneous message when processing message M_i,
- P_{XmNDb} the probability of no detection when using m distinct error checking functions in the case of a set of b messages,

where $X \in \{S, M\}$ specifies the error checking scheme, where S stands for SECF and M for MECF.

In the case of SECF, all messages are checked using the same checking function F. As M_1 is not detected as erroneous (with probability p_{SND1}), then the conditional probability of no detection for each subsequent message (p_{SNDi}, i = 2,..., b) is equal to 1 when the part of the frame that is meant to be protected by the end-to-end checking function remains unchanged. When linear codes are being used, this probability is also equal to 1, even if the whole set of protected information is not kept constant, provided that the very part of the message that is affected by the error is kept unchanged. Such a behavior could be caused by a permanent fault affecting an interstage node and would result in the same (repetitive) error. Indeed, in such a context, as the error is not detected when processing M_1 (i.e., the error is a multiple of the underlying polynomial generator), there is no chance for it to be detected when processing subsequent messages due to the linear property of the code. Accordingly, it can be conservatively assumed that the probability of no detection for a set of b messages P_{SNDb} is governed by p_{SND1}.

Now, let us consider MECF. We propose to use m distinct checking functions for a set of b messages and to recurrently change them according to a given scheduling policy. The policy we consider herein is described by:

$$F_1, F_2,..., F_m, F_1, F_2,..., F_m, \text{ etc.} \quad (1)$$

Note that for sake of simplicity, we assume first that $m = b$. Other policies were also investigated (see [12] for details).

As for SECF, p_{MND1} designates the probability of no detection when processing message M_1. Since the $m = b$ checking functions used in MECF provide complementary error detection capabilities (see Section 5 for details), the successive probabilities of no detection (p_{MNDi}, i = 2,..., b) can be considered as independent, complementary and much lower than 1. Accordingly:

$$P_{MbNDb} = p_{MND1} \times p_{MND2} \times ... \times p_{MNDb}. \quad (2)$$

Thus, the MECF scheme somewhat "cumulates" the error detection capabilities of the m checking functions.

For sake of consistency, it is appropriate to assume first that function F (SECF) and functions F_i (MECF) provide intrinsically the same error detection capabilities. In particular, this means that $p_{SND1} = p_{MND1} = p_{ND1} \approx 2^{-c}$ with respect to multiple errors, where c designates the number of checking bits of the various codes.

Accordingly, for SECF, $P_{SbNDb} \approx 2^{-c}$. The only possibility to increase the error detection capability for a set of messages is to increase the number of check bits.

For MECF, assuming "full complementarity" among the checking functions would lead to $P_{MbNDb} \approx (2^{-c})^b$; this means that the integrity over a set of messages can be increased significantly by using moderate size codes for each checking function. Actually, this is equivalent to a single code featuring b x c check bits.

In practice, one may want to consider a fewer number of functions than the size of the set ($m < b$). In that case, the probability for a repetitive error affecting b messages to remain undetected is governed by the number of distinct functions, i.e., $P_{MmNDb} \approx (2^{-c})^m$. Actually, in that case $p_{MNDj} \approx 1$ for $m < j \leq b$ (as for SECF).

Thus, compared to the use of a single checking function with m x c check bits, instead of being detected when processing M_1, it might occur that the detection of the (repetitive) error would be delayed until the application of the mth function F_m, i.e., when processing message M_m. However, this is compensated by the fact that much less control bits would be appended to each message. Indeed, due to the slow dynamics of the considered class of systems, there is no need to ensure a error detection for each message, thus, MECF represents a solution satisfying both the integrity objective and the requirements for restricting the level of redundancy.

One may find some analogy with the principle underlying the "Turbo Codes" identified in Section 3.3. However, two main differences exist: i) we do not consider data interleaving[4], ii) we apply m distinct checking functions to m *successive* messages, while the basic Turbo Codes apply 2 functions to *each* message.

4.2 Impact of the recovery strategies

The MECF scheme achieves a high power of detection of errors affecting a set of successive messages, thus ensuring high communication integrity between the control nodes and the controlled nodes handling the flight control surfaces. Nevertheless, the avoidance of the undesired event (namely, the runaway of some control surface) requires that the number of undetected erroneous messages delivered within a set of b messages remains lower than a prescribed threshold.

[4] The mathematical foundations that govern interleaving and that would help choosing an interleaving function are still to be studied. Conversely, a rational basis exists for selecting checking functions for a CRC implementation of MECF (see Section 5).

Clearly, the use of either of the recovery strategies introduced in Section 2.1.2 might result in a different impact on the risk of a runaway. Indeed, the considered recovery strategies are meant to maintain a certain level of availability for the controlled servomechanism. However, delaying the triggering of the recovery action may have an impact on safety (by maintaining the delivery of an erroneous reference command whose accumulation might lead to a runaway).

The aim of this section is to define the characteristics of the preferred recovery strategy, i.e., that minimizes: i) the risk of occurrence of a runaway and ii) the recovery latency that we define as the time elapsed between the first reception of an erroneous message (which could be detected or not) and the initiation of the recovery action. Indeed, as usual, the shortest the recovery latency, the most efficient the recovery action.

Figure 3 describes the behaviors induced by the two instantiations of the recovery strategies considered in Section 2.1.2, RS1(3) and RS2(3,10), when considering MECF with increasing values for the numbers of error checking functions (m = 1, 2, 3, and 4). The basic cyclic policy — see expression (1) — is used for scheduling these functions. In order to support our analysis, we consider scenarios in which the considered faults induce repetitive erroneous messages that remain steadily undetected by a specific checking function (F_1, for sake of simplicity). Also, it is worth pointing out that the behaviors observed for m = 1 are obviously similar for the two recovery strategies.

The first main observation is that, whatever the recovery strategy, the greater the number of checking functions, the shorter the Recovery Latency (**RL**). Indeed, for the RS2 (3,10) strategy, this duration steadily decreases from 50 ms for m = 2 to 30 ms for m = 4. However, it is worth noting that the RL remains unchanged and equal to 30 ms when $m \geq 4$ for both recovery strategies. This explains our choice to bound the analysis reported here to a cyclic policy of 4 checking functions in the case when r = 3.

The second main observation concerns the identification of the preferable recovery strategy. Using RS2(3,10), the risk of a runaway can be eliminated[5] for m = 2, while for RS1(3), 4 functions are necessary to eliminate this risk and also to achieve the same value for the RL as the one obtained for RS2(3,10). Accordingly, for a fixed value of parameter m (i.e., the number of error checking functions), RS2 is best adapted to the case study, for the two following reasons:

1) It ensures the lowest risk of occurrence of the UE, which is the primary objective considering the high integrity requirement to ensure a rate of occurrence of the UE lower than 10^{-9}/h.
2) It leads to the lowest RL and thus to a better availability of the FCS.

5 Implementation Using CRC

In practice, various types of detecting codes could be used to implement the MECF scheme. In this section, we describe an implementation based on CRC. Prior to describing this implementation, we briefly present two analyses that substantiate the choice of CRC. More details can be found in [12].

[5] Actually, the probability associated to this risk is lower than the prescribed threshold.

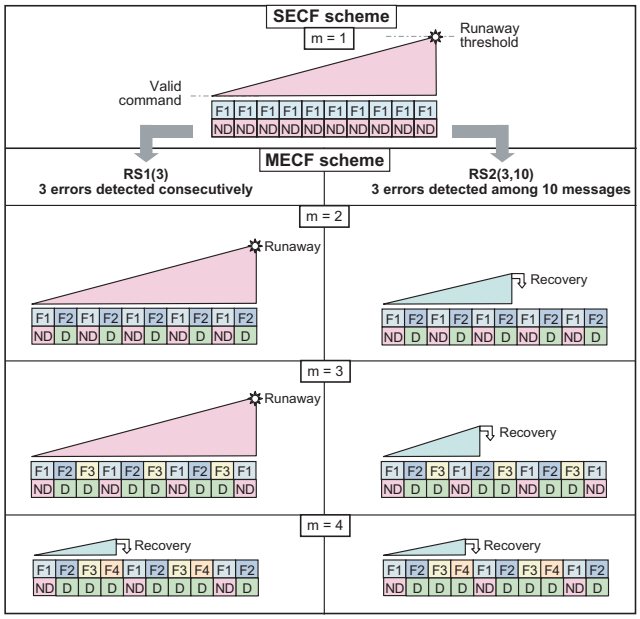

Figure 3: Comparison of RS1(3) and RS2(3,10)

5.1 Some rationales for using CRC

We successively address two important items:

- complementarity of the error detection capabilities of the checking functions to be included into the MECF scheme,
- compatibility of the processing (coding and decoding) of the checking functions at application level with respect to the temporal constraints of the targeted application.

5.1.1 Complementarity of detection capabilities

Since MECF involves several checking functions, it is necessary to propose a sound basis to substantiate the choice of functions that optimize the complementary of the detection capabilities procured by each function.

In the case of CRC, the complementarity can be based on the consideration of the properties of the generator polynomials G(x) that underlie the respective codes (see [2], p. 161). More precisely, the specific relevant property is exhibited by considering the decomposition of the generator polynomials into the product of irreducible polynomials. Let us recall briefly that a polynomial is considered as irreducible, if the greatest common divisor of its coefficients is 1[6].

All the CRC coded words generated by polynomial G(x) are inevitably multiple of it and thus contain the irreducible factors of this polynomial. Accordingly, all errors that remain undetected by such a code are necessarily multiple of its generator polynomial. Thus, for another code generated by a distinct polynomial to exhibit the highest (complementary) error detection capability, it is necessary that the two polynomials share the least irreducible polynomials. Accordingly, the examination of the features of the underlying polynomials, according to the properties of their irreducible factors, provides a strong and pragmatic basis to support the decision whether to include a specific set of CRC codes in the MECF scheme. Similar arguments were used in [21], albeit in a slightly different context.

[6] An analogy can be made to a prime number that is divisible only by 1 and itself.

5.1.2 Compatibility with temporal constraints

For an implementation of the MECF scheme using CRC to be compatible with the temporal constraints, the computation time necessary to the generation and checking of the CRC bits should be small with respect to the cycle time of the control system (10 ms in our case).

The corresponding evaluation has consisted in the determination of the computation time for three algorithms (Cyclic Redundancy Code Bitwise, Cyclic Redundancy Code Table lookup, Cyclic Redundancy Code Reduced Table lookup) corresponding to classical software implementations of CRC [24]. For the analysis, we have considered the case of 128-bit data and 16-bit CRC size and a microcomputer representative of those used on-board commercial airplanes (PIC18C542 at 20Mhz).

Our analyses show that the computation times range between 35 µs and 90 µs, depending on the algorithm. These times obtained with computers of modest performance are less than 1% of the cycle duration. Accordingly, it can be confidently concluded that a software implementation of the CRC codes is really compatible with the temporal constraints. Recent work [25] proposes useful hints to speed up computational issues.

5.2 CRC-based implementation of MECF

In the case of CRC, the MECF scheme consists in modifying cyclically (i.e., according to the scheduling policy we consider in this paper — see also (1)) the generator polynomials G_1, G_2,..., G_m for the messages transmitted. As discussed previously, the overall contribution in terms of error detection capability depends on the choice of these polynomials.

We specify in the next section the characteristics of the generator polynomials composing the MECF scheme. For this, we analyze the correlation that exists between the complementarity of their intrinsic errors detection capabilities and the resulting efficiency of MECF scheme they compose. Then, we present the strategy allowing to select or to derive these polynomials.

5.2.1 Generator polynomials characteristics

The generator polynomials to be considered for MECF must possess two essential characteristics:

- significant intrinsic error detection capability for each message,
- complementarity between their errors detection capabilities with respect to a set of messages.

The complementarity between the errors detection capabilities of the m functions used is all the more significant as the m generator polynomials on which they are based are such as the degree of the product of their common factors is minimal.

Let us denote Π, the polynomial with smaller degree that satisfies a set of "generic" or "standard" properties in terms of error detection. Concretely, the polynomial Π will have to satisfy the properties of the CRC standard codes concerning the detection of the following error types:

- all single errors, if the coefficient of x^0 of $G(x)$ equals 1,
- all double errors, if $G(x)$ contains a primitive polynomial with a maximum of 3 terms,
- all odd weight errors, if $G(x)$ contains the factor $(1+x)$.

Thus, to detect these three types of errors, and to have smallest degree, the polynomial Π should be equal to the product of the two primitive polynomials $\Pi_1 = 1+x$ and $\Pi_2 = 1+x+x^2$, i.e., $\Pi = 1+x^3$.

In what follows, we describe the approach leading to the identification of suitable generator polynomials that possess the double characteristics of intrinsic standard detection capabilities and complementary detection capabilities.

5.2.2 Selection of the generator polynomials

Figure 4 illustrates the process for selecting the generator polynomials that are good candidates to compose a MECF scheme. It is assumed that generator polynomial G_1 is decomposable into five irreducible polynomials: i) Π_1 and Π_2 that form a polynomial Π satisfying the generic properties of the majority of standard CRC, e.g., as indicated in Section 5.2.1 and ii) three additional irreducible polynomials denoted P_1, P_2 and P_3 of distinct degrees.

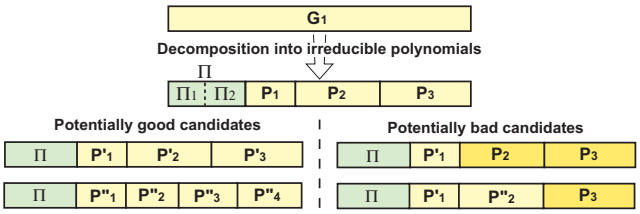

Figure 4: Selection of generator polynomials candidates

The figure exemplifies potentially good and bad candidates for the selection of generator polynomials, on the basis of polynomial G_1. A judicious choice corresponds to a polynomial including Π and one or several irreducible polynomials all distinct from P_1, P_2 and P_3. A bad candidate would correspond to a polynomial including one or several irreducible polynomials that already compose G_1.

Table 1 provides examples of generator polynomials for 16-bit CRC that would constitute good and bad candidates for the MECF scheme.

All polynomials shown are decomposable into Π and two other factors that are irreducible polynomials of degrees 7 and 8. Here Π is simply $1+x$. Good candidates shown include factors that are all distinct from those of G_1, while the bad candidates share the same polynomial of degree 7.

6 Validation of the MECF Scheme

The goal of the validation studies conducted is twofold:

- confirm the guidelines proposed for preserving the complementarity with respect to error detection capabilities of the codes participating to the MECF scheme,
- assess the improvement in integrity gained from the application of the MECF scheme in the case of the target FCS.

The validation is based on simulation models of the MECF scheme developed in the Matlab-Simulink environment. Figure 5 sketches the simulation framework using the Matlab-Simulink syntax. The diagram illustrates the process for assessing the complementarity with respect to error detection capabilities between two generator polynomials (namely, two standard polynomials CRC-16 and IEEE WG77.1 — see also Table 2). CRC-16 is considered as the reference polynomial (denoted $\mathbf{G_1}$ in Figure 4) and IEEE WG77.1 as a candidate polynomial.

Table 1: Generator polynomials for 16-bit CRC

$G_1(x) = \underline{(1+x)} \cdot \underline{(1+x+x^7)} \cdot (1+x^2+x^3+x^4+x^8) = 1+x^3+x^5+x^6+x^7+x^9+x^{10}+x^{12}+x^{15}+x^{16}$		
Examples of potentially good candidates **$G(x)$ = (1+x) . 7-degree irreducible polynomial . 8-degree irreducible polynomial**		
Identifier	Polynomial representation	Decomposition into irreducible polynomials
$G_2(x)$	$1+x+x^6+x^7+x^8+x^9+x^{10}+x^{13}+x^{15}+x^{16}$	$\underline{(1+x)} \cdot (1+x+x^3+x^5+x^7) \cdot (1+x+x^2+x^4+x^5+x^6+x^8)$
$G_3(x)$	$1+x+x^6+x^{10}+x^{12}+x^{16}$	$\underline{(1+x)} \cdot (1+x+x^2+x^3+x^7) \cdot (1+x+x^4+x^5+x^6+x^7+x^8)$
$G_4(x)$	$1+x^5+x^6+x^7+x^8+x^9+x^{10}+x^{16}$	$\underline{(1+x)} \cdot (1+x^3+x^7) \cdot (1+x+x^2+x^5+x^6+x^7+x^8)$
Examples of potentially bad candidates **$G(x)$ = (1+x) . (1+x+x^7) . 8-degree irreducible polynomial**		
$G_5(x)$	$1+x+x^2+x^3+x^5+x^6+x^9+x^{10}+x^{12}+x^{14}+x^{15}+x^{16}$	$\underline{(1+x)} \cdot \underline{(1+x+x^7)} \cdot (1+x+x^5+x^6+x^8)$
$G_6(x)$	$1+x^3+x^6+x^7+x^{10}+x^{13}+x^{14}+x^{16}$	$\underline{(1+x)} \cdot \underline{(1+x+x^7)} \cdot (1+x^2+x^3+x^4+x^5+x^7+x^8)$

To optimize the duration of the simulations and better exemplify the impact of the proposed scheme, only (erroneous) messages that are multiple of the reference generator polynomial are considered. This convincingly models the case of repetitive errors that are undetected by the classical application of CRC using a single error checking function (i.e., the SECF scheme).

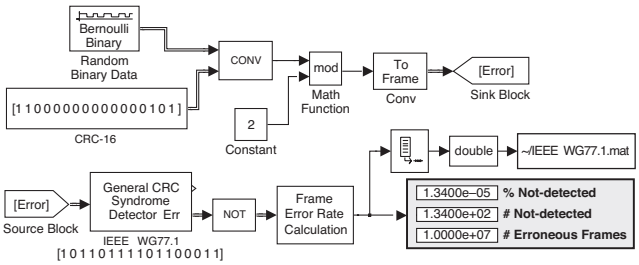

Figure 5: The simulation framework

The top part of the diagram describes the generation of (erroneous) message frames that are multiple of the reference polynomial (CRC-16 in this case). Message frames of 100 bits are being used. The lower part describes the process for checking these erroneous frames using a "General CRC Syndrome Detector" based on the candidate polynomial (IEEE WG77.1). It also identifies and cumulates the cases when no detection occurs. The data displayed are: i) the percentage of errors not detected, ii) the number of errors not detected, iii) the total number of erroneous frames. For each candidate polynomial, the simulation runs consisted in the injection of 10^7 (erroneous) message frames multiple of the reference polynomial. The results obtained are stored into a file (~/IEEE WG77.1.mat) for further processing and analysis.

Such a process has been used to test several pairs of polynomials. This addresses specifically the assessment of the MECF for $m = 2$. A simple extension of this process has been used to test the MECF scheme for $m > 2$ (e.g., see [12]).

6.1 Analysis of code complementarity

Two analyses are considered that successively study the proposed criteria for discriminating good and bad generator polynomials for contributing to a MECF scheme and study whether the selection of standard or purposely designed (custom) polynomials has an impact on the error detection capability of the MECF scheme.

6.1.1 Good *vs.* bad polynomial candidates

The analysis builds up on the detection complementarity with respect to G_1 procured by the polynomials of Table 1. It is based on the comparison of simulation results obtained with the estimated theoretical objective. Polynomials G_2, G_3 and G_4 (good candidates) differ from G_1 by a polynomial factor of degree 15. Thus, one would expect that the theoretical probability that erroneous messages (multiple of G_1) are also undetected by G_2, G_3 or G_4 be equal to $2^{-15} \approx 3 \times 10^{-5}$. The probabilities obtained via simulation are: 3.05×10^{-5} for G_2, 2.96×10^{-5} for G_3, and 2.81×10^{-5} for G_4.

Table 2: Good candidate generator polynomials for CRC-16

$G_a(x) = \underline{(1+x)} \cdot (1+x+x^{15}) = 1+x^2+x^{15}+x^{16}$ — **Standard generator polynomial: CRC-16**		
Standard generator polynomials **$G(x)$ = (1+x) . 15-degree polynomial**		
Identifier	Polynomial representation	Decomposition into irreducible polynomials
$G_b(x)$: IEEE-WG77.1	$1+x+x^5+x^6+x^8+x^9+x^{10}+x^{11}+x^{13}+x^{14}+x^{16}$	$(1+x^2+x^3+x^4+x^8) \cdot (1+x+x^2+x^4+x^5+x^6+x^8)$
$G_c(x)$: CRC-CCITT	$1+x^5+x^{12}+x^{16}$	$\underline{(1+x)} \cdot (1+x+x^2+x^3+x^4+x^{12}+x^{13}+x^{14}+x^{15})$
$G_d(x)$: IBM-SDLC	$1+x+x^2+x^4+x^7+x^{13}+x^{15}+x^{16}$	$\underline{(1+x)}^2 \cdot (1+x+x^3+x^4+x^5+x^6+x^8+x^{10}+x^{12}+x^{13}+x^{14})$
$G_e(x)$: CRC-16Q*	$1+x+x^3+x^4+x^5+x^6+x^8+x^{11}+x^{15}+x^{16}$	$\underline{(1+x)} \cdot (1+x^3+x^5+x^8+x^9+x^{10}+x^{15})$
$G_f(x)$: IEC-TC57	$1+x+x^4+x^7+x^8+x^9+x^{11}+x^{12}+x^{14}+x^{16}$	$\underline{(1+x)}^2 \cdot (1+x+x^3+x^6+x^7) \cdot (1+x^2+x^3+x^4+x^5+x^6+x^7)$
Custom generator polynomials **$G(x)$ = (1+x) . 7-degree irreducible polynomial . 8-degree irreducible polynomial**		
$G_g(x) = G_3(x)$	$1+x+x^6+x^{10}+x^{12}+x^{16}$	$\underline{(1+x)} \cdot (1+x+x^2+x^3+x^7) \cdot (1+x+x^4+x^5+x^6+x^7+x^8)$
$G_h(x) = G_4(x)$	$1+x^5+x^6+x^7+x^8+x^9+x^{10}+x^{16}$	$\underline{(1+x)} \cdot (1+x^3+x^7) \cdot (1+x+x^2+x^5+x^6+x^7+x^8)$

For polynomials G_5 and G_6 (bad candidates) that only differ from G_1 by a polynomial factor of degree 8, one would expect that the theoretical probability that erroneous messages (multiple of G_1) are also undetected by G_5 or G_6 be equal to $2^{-8} \approx 3.9 \times 10^{-3}$. The figures obtained via simulation are respectively equal to 3.89×10^{-3} and 3.91×10^{-3}.
These analyses show a good match between the experimental and theoretical results.

6.1.2 Standard *vs.* custom polynomials

We consider eight generator polynomials for the analyses (Table 2): six are standard ones and two are "custom" polynomials derived on the basis of the procedure sketched in Section 5.2.2. The "reference" polynomial (G_a), with respect to which the complementary error detectability provided by the other codes is being evaluated, is the standard generator polynomial CRC-16. All these polynomials are potential good candidates for MECF, according to the complementarity criteria.

Figure 6 plots the ratio of erroneous messages (multiple of G_a) not detected by the other polynomials in Table 2 when paired to G_a, as a function of the number of pseudo-randomly generated erroneous messages. To facilitate the readability, the figure only shows the results for three standard generator polynomials (namely, G_b, G_c and G_d), in addition to the two custom polynomials G_g and G_h. The simulation results show that the ratio of undetected errors for the $[G_a - G_b]$ pair converges to the theoretical value of 2^{-16} ($\approx 1{,}5 \times 10^{-5}$), while for all the other pairs is converges to the theoretical value of 2^{-15} ($\approx 3 \times 10^{-5}$). These results reflect the differences between G_b and the other polynomials: all polynomials G_α, with $\alpha \in \{c, d, e, f, g, h\}$ share polynomial $(1+x)$ with G_a, while G_b has no polynomial in common with G_a. These results fully support the validity of the criteria put forward for optimizing the complementarity of error detection by CRC code candidates for the MECF scheme

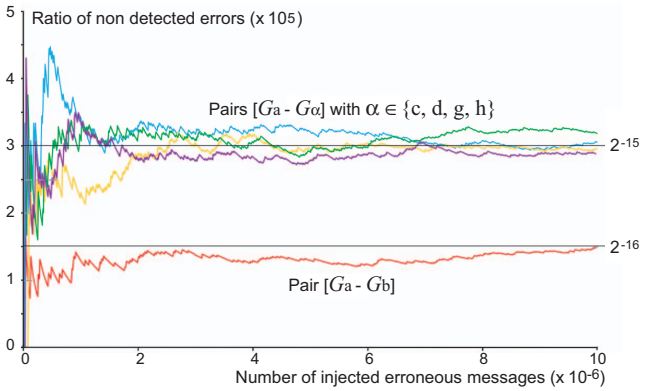

Figure 6: No detection for various G_a-related pairs

The results indicate also that the error detection capabilities for the schemes pairing G_a, either a standard or custom code are equivalent. More specifically, from the 10^7 simulation runs featuring (erroneous) messages multiple of G_a, the number of errors not detected, when pairing G_a with either a standard or a custom polynomial, were very similar:

- G_a and standard: $[G_a - G_c] = 318$; $[G_a - G_d] = 286$; $[G_a - G_e] = 294$; $[G_a - G_f] = 292$;
- G_a and custom: $[G_a - G_g] = 296$; $[G_a - G_h] = 281$.

The figure also reveals a significantly different behavior for the $[G_a - G_b]$ pair, for which the number of non detections is only 146 (about half of the figures observed for the other pairs).

6.2 Assessment of the MECF scheme

We summarize here the main results of the risk analysis that we have conducted for the FCS (see [12]). The objective is to assess the gain procured by the application of the cyclic scheduling MECF scheme with respect to the classical application of CRC, i.e., the SECF scheme.
The focus is put on the study of the functional failures affecting the interstage nodes between main control nodes and the controlled nodes associated to the servo-mechanisms of the flight surfaces, because it was shown that they had the most severe impact on the risk of occurrence of the undesired event (control surface runaway). Let us recall that the target is to maintain the rate of this risk much lower than 10^{-9}/h. As discussed in Section 2.2, we assume a failure rate of about 10^{-7}/h for repetitive errors induced by an interstage node. More specifically, we focus here on two fault classes affecting such nodes: i) stuck-at faults affecting the buffers handling the messages and ii) stuck-at faults on buffer address control.
Table 3 shows the risks induced by these failure modes for several protection schemes, when considering 16-bit CRCs, namely SECF and MECF (for m = 2 and 3), for which the considered CRC groupings are respectively $[G_a - G_b]$ and $[G_a - G_b - G_c]$ from Table 2.

Table 3: Rate of occurrence of UE (control surface runaway)

Protection schemes → / ↓ Fault classes: stuck-at on	SECF	MECF	
	(m = 1)	m = 2	m = 3
- buffer memory	1.5×10^{-12} / h	2.3×10^{-17} / h	6.9×10^{-22} / h
- address control	10^{-7} / h	1.5×10^{-12} / h	4.5×10^{-17} / h

For SECF, the figures in the table show that, for the first fault class, the simple use of a 16-bit CRC code leads to a risk of 1.5×10^{-12}/h, thus much lower than the threshold of 10^{-9}/h. This is no longer the case for the second fault class that induces repetitive errors for which no protection can be obtained by means of a classical CRC coding scheme (either at communication layer or at application layer). This is why the related risk was estimated to be equal to a fraction of the corresponding failure rate of the node. Applying the MECF scheme significantly reduces the rates and allows for meeting the integrity target. For example, for the second fault class, it is decreased to 1.5×10^{-12} / h. This rate is further reduced when using three error checking functions to 4.5×10^{-17}/h. It is also noteworthy that, while the integrity objective was already satisfied by SECF for the first fault class, the application of MECF significantly reduces the risk figure.
The rates for MECF are confidently obtained thanks to the match observed between the simulation and theoretical results. For example, the figure for the first fault class for MECF [m = 2] (2.3×10^{-17}) is obtained as the product of the related figure for the SECF scheme by the detection miss for the $[G_a - G_b]$ pair (i.e., $1.46 \times 10^{-5} \approx 2^{-16}$) as shown in Figure 6. The fact that G_c shares the factor $(1+x)$ in common with G_a, means that the detection miss to be considered between MECF [m = 2] and MECF [m = 3] is about $3 \times 10^{-5} \approx 2^{-15}$.

These results clearly show the benefit gained from the application of the MECF scheme we have proposed and thus substantiate the claim that such a scheme is able to maintain the rates of control surface runaway in-line with the stringent integrity level assigned.

7 Concluding Remarks

The development of critical control systems, such as those considered in this paper that aim at supporting the deployment of fully digital devices for future flight control systems (FCS), rely increasingly on the use of fieldbus networks. In this context, we have focused our attention on the communication integrity procured by this type of networks. The analysis of the risks of suitable networks has shown that the major difficulty was to cope with repetitive errors that might be induced by the failures of complex interstage nodes.

In order to achieve the stringent integrity level assigned to the application, we have proposed an original approach that departs from the classical end-to-end checking approaches and solutions relying on redundant topologies for ensuring communication integrity. Indeed, the specific features of the class of control systems considered do not ask for a high detection strength for every message, but rather for a set of messages. We have taken advantage of this to devise a solution able to achieve the assigned integrity at reduced cost. The scheme we propose consists in using distinct error checking functions for the transmission of successive messages. This way, the detection capabilities of each function can be cumulated to cope efficiently with repetitive errors.

In this paper, we have also shown that cyclic redundancy codes (CRC) are particularly well suited for implementing our proposal. Actually, elaborating on the mathematical foundations on which these codes rely, it has been possible to identify checking functions that best exhibit complementary error detection capabilities with respect to repetitive errors. The basic guideline is to select functions such as the degree of the product of their common irreducible polynomials is minimal. Two main results surface from the validation studies: i) the confirmation of the criteria proposed for selecting the checking functions and ii) the evidence that thanks to the proposed solution, the rate of occurrence of the undesired event (runaway of the flight control surfaces) can be kept at a value compatible with the assigned target, i.e., lower than 10^{-9}/h.

Acknowledgements

This work was supported in part by Contract Airbus — LAAS-CNRS "Integrity of Communications for Advanced Flight Control Systems". The authors would like to thank the anonymous reviewers for their helpful comments and Geert Deconinck for his constructive feedback while "shep-herding" the production of the final version of the paper.

References

[1] A. Avizienis, J-C. Laprie, B. Randell and C. Landwehr, "Basic Concepts and Taxonomy of Dependable and Secure Computing," *IEEE Trans. on Dependable and Secure Computing,* vol. 1, no. 1, pp. 11-33, Jan.-March 2004.

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

[3] P. Koopman and T. Chakravarty, "Cyclic Redundancy Code (CRC) Polynomial Selection For Embedded Networks," *Proc. IEEE/IFIP DSN-2004,* Florence, Italy, 2004, pp. 145-154.

[4] M. Paulitsch, J. Morris, B. Hall, K. Driscoll, E. Latronico and P. Koopman, "Coverage and the Use of Cyclic Redundancy Codes in Ultra-Dependable Systems," *Proc. IEEE/IFIP DSN-2005,* Yokohama, Japan, 2005, pp. 346-355.

[5] *Software Considerations in Airborne Systems and Equipment Certification*, RTCA and EUROCAE Standard Document no. DO178B/ED-12B, 1992.

[6] *Certification Considerations for Highly-Integrated or Complex Aircraft Systems*, SAE Standard Document no. SAE ARP 4754, 1996.

[7] R. C. Hammett and P. S. Babcock, "Achieving 10(-9) Dependability with Drive-by-Wire Systems," *Proc. SAE World Congress,* Detroit, MI, USA, 2003. (Draper Report P-4027).

[8] C. Wilwert, Y. Song, F. Simonot-Lion and T. Clément, "Evaluating Quality of Service and Behavioral Reliability of Steer-by-Wire Systems," *Proc. ETFA'03,* Lisbon, Portugal, 2003, pp. 193-200, (IEEE CS Press).

[9] *Functional Safety of Electrical/Electronic/Programmable* IEC Standard Document no. 61508-3 - First edition, 1998.

[10] H. Kopetz, *Real-Time Systems - Design Principles for Distributed Embedded Applications*, Springer, 1997.

[11] H. T. Brodtkorp, *A Safety Layer for Foundation Fieldbus,* PhD Dissertation, University of Oslo, Norway, 2001.

[12] A. Youssef, *High Integrity Communication Networks for Critical Control Systems Integrating Microsystem Arrays,* PhD Dissertation, INP Toulouse, France, 2005. (In French).

[13] H. Kopetz, "A Comparison of CAN and TTP," *Proc. 15th IFAC Workshop on Distributed Computer Control Systems (DCCS-98),* Como, Italy, 1998, pp. 117-128.

[14] D. Powell (Ed.) *A Generic Fault-Tolerant Architecture for Real-Time Dependable Systems,* Kluwer, 2001.

[15] K. Hoyme and K. Driscoll, "SAFEbus™," *IEEE Aerosp. & Electr. Syst. Mag.*, vol. 8, no. 3, pp. 34-39, March 1993.

[16] P. S. Miner, M. Malekpour and W. Torres, "A Conceptual Design for a Reliable Optical Bus (ROBUS)," *Proc. IEEE DASC-21,* Hampton, VA, USA, 2002, pp. 13D3-1-13D3-11.

[17] A. Grnarov, L. Kleinrock and M. Gerla, "A Highly Reliable Distributed Loop Network Architecture," *Proc. IEEE FTCS-10,* Kyoto, Japan, 1980, pp. 319-324.

[18] B. Hall, K. Driscoll, M. Paulisch and S. Djani-Brown, "Ringing Out Fault Tolerance. A New Ring Network for Superior Low-Cost Dependability," *Proc. IEEE/IFIP DSN-2005,* Yokohama, Japan, 2005, pp. 298-307.

[19] H. Jitsukawa and T. Maruyama, "Method of Error Detection and Correction by Majority," US Patent no. 4670880, 1987, available from www.freepatentsonline.com.

[20] S. Mitra and Edward J. McCluskey, "Design of Redundant Systems Protected Against Common-Mode Failures," *Proc. IEEE VTS 2001,* Marina del Rey, CA, USA, 2002, pp. 190-197.

[21] J. E. Mazo and B. R. Saltzberg, "Error-Burst Detection with Tandem CRCs," IEEE Transactions on Communications, vol. 39, no. 8, pp. 1175-1178, August 1991.

[22] C. Berrou and A. Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo-Codes," *IEEE Trans. on Comm.*, vol. 44, no. 10, pp. 1261-1271, 1996.

[23] J. Aidemark, P. Folkesson and J. Karlsson, "A Framework for Node-Level Fault Tolerance in Distributed Real-time Systems," *Proc. IEEE/IFIP DSN-2005,* Yokohama, Japan, 2005, pp. 656-665.

[24] T. V. Ramabadran and S. S. Gaitonde, "A Tutorial on CRC Computations," *IEEE Micro*, vol. 8, no. 4, pp. 62-75, 1988.

[25] J. Ray and P. Koopman, "Efficient High Hamming Distance CRCs for Embedded Networks," *Proc. IEEE/IFIP DSN-2006*, Philadelphia, PA, USA, 2006, pp. 3-12.