Assessment of Message Missing Failures in FlexRay-Based Networks*

Vahid Lari, Mehdi Dehbashi, Seyed Ghassem Miremadi, Navid Farazmand
Sharif University of Technology
{Lari, Dehbashi, Farazmand}@ce.sharif.edu
Miremadi@sharif.edu

Abstract

This paper assesses message missing failures in a FlexRay-based network. The assessment is based on about 35680 bit-flip fault injections inside different parts of the FlexRay communication controller; the parts are: controller host interface, protocol operation control, coding and decoding unit, media access control and clock synchronization process. To do this, a FlexRay communication controller is modeled by Verilog HDL at the behavioral level. This HDL model of the controller is exploited to setup a FlexRay-based network composed of four nodes. The results of fault injection show that about 35% of faults led to the message missing failures. The controller host interface and the clock synchronization process of the FlexRay were the most sensitive parts to the message missing failures. The coding and decoding unit of the FlexRay was the least sensitive part to these failures.

1. Introduction

Safety in distributed systems such as automotive systems and avionics is of decisive importance due to system failures which may threat human life. In a distributed system, each node consists of three parts [1]: 1) I/O part, 2) host part, and 3) communication controller.

In general, communication activities can be triggered either dynamically, in response to an event (event-triggered), or statically, at predetermined moments in time (time-triggered). Examples of time-triggered protocols are the SAFEbus, SPIDER, and Time-Triggered Protocol (TTP). The main drawback of the time-triggered protocols is their lack of flexibility [2]. Examples of event-triggered protocols are the Byteflight introduced by BMW Company for automotive applications, CAN, LonWorks and Profibus. The main drawback of the event-triggered protocols is their lack of predictability. A large consortium of automotive manufacturers and suppliers has proposed a hybrid type of protocol, namely, the FlexRay communication protocol [3]. The FlexRay allows the sharing of the bus among event-triggered and time-triggered messages, thus offering the advantages of both protocols. It is reported that the FlexRay will very likely become the de-facto standard for in-vehicle communications [2] [4]. The FlexRay defines a communication cycle (bus cycle) as the combination of a time-triggered (or static) window, an event-triggered (or dynamic) window, a symbol window and a network idle time (NIT) window. The time-triggered window is similar to TTP, and employs a time-division multiple-access (TDMA) mechanism. The event-triggered window of the FlexRay protocol is similar to Byteflight protocol and uses a flexible TDMA (FTDMA) bus access method. The symbol window is a communication period in which a symbol can be transmitted on the network. The NIT window is a communication-free period that specifies the end of each communication cycle.

The importance of safety in critical distributed applications signals to pay specific attention to the reliability of communication protocols. One way to assess the reliability of communication protocols is by fault injection. In [5], a simulation-based fault injection has been used for the assessment of message missings in the CAN protocol. Effects of masquerade failures have been investigated using a simulation-based fault injection in the CAN protocol [6]. Evaluation of TTP/C communication controller by heavy-ion fault injection (hardware-based fault injection) has been performed in [7]. The purpose of the experiments in that paper was to validate the fail silence property of the TTP/C by injecting faults in a single node. The relationship between the number of nodes in a cluster and the slightly-off-specification (SOS) failures has been

* This work was partially supported by a grant from Iran Telecommunication Research Center (ITRC).

0-7695-3054-0/07 $25.00 © 2007 IEEE
DOI 10.1109/PRDC.2007.6

assessed using heavy-ion fault injection [8]. In [9], the TTP/C protocol with bus and star topologies has been investigated using SWIFI fault injection. Here, the effects of the SOS failures in the bus and star topologies with respect to the start of frame transmission have been studied. In [10] [11], a generic tool was developed for monitoring and diagnosis of a FlexRay-based system as well as for a CAN-based system. This tool has been used by the FlexRay consortium to perform extended fault injection for evaluating of the FlexRay communication protocol. One important limitation of this tool is that faults cannot be injected inside different parts of the FlexRay protocol.

This paper assesses the message missing failures by injecting 35680 bit-flip faults inside different parts of the FlexRay protocol. The assessment is based on faults which disturb the message sending and message receiving in a FlexRay-based network. To do this, a FlexRay communication controller is modeled by Verilog HDL at the behavioral level. This HDL model of the controller is exploited to setup a FlexRay-based network composed of four nodes. The results of fault injection can be divided into two main categories: 1) the faults resulting in the message missing failures in the FlexRay network are assessed. Here, the message missing failures that occurred in time-triggered and event-triggered window of the FlexRay communication cycle are evaluated. Also, the sensitive points of the FlexRay protocol to the message missing failures are identified; 2) the faults resulting in the three kinds of errors, namely, content errors, syntax errors and boundary violation errors are characterized. Here, the most sensitive and the less sensitive points of the FlexRay protocol to faults are identified.

This paper is organized in five sections. Section 2 presents the message missing failures and error models found in the FlexRay protocol. The experimental setup is given in section 3, and the results are presented in section 4. The last section concludes the work.

2. Error model and message missing failure

Error models: The FlexRay protocol has different mechanisms for detecting errors in the controller. At the end of each time slot, frame and symbol process (FSP) part checks the presence of any error in that slot and informs the host about it. This protocol defines three main errors that can occur in each slot: Syntax error, content error and boundary violation error. The syntax error denotes the presence of a syntactic error in a time slot, the content error denotes the presence of an error in content of a received frame and boundary violation error denotes whether a boundary violation occurred at boundary of the corresponding slot.

Message missing failures: Faults can disturb sending or receiving of a message in a node of a distributed system and cause a message to be missed. In this paper the message missing failures are assessed in two aspects:

1- Because of a fault in the communication controller of the sender node, it does not send its message.
2- Because of a fault in the communication controller of the sender node, it sends its message incorrectly on the network, thus, the message won't be accepted in receiver nodes.

In this experiment we assumed that the host is fast enough to generate the messages for sending, and to read the messages from communication controller. Meanwhile, the number of generated messages by the host is exactly equal to the number of the IDs that has been allocated to that host.

3. Experimental setup

In order to perform an experiment on the FlexRay controller a network consisting of nodes that have this controller should be set up. So, a model of FlexRay controller has been implemented at behavioral level according to the FlexRay protocol specification [3]. This controller has been implemented by hardware description language, Verilog, and Modelsim 6.1 simulator. This FlexRay controller has been tested according to the FlexRay protocol conformance test specification [12].

The implemented controller has usual capabilities of FlexRay protocol such as sending and receiving the static and dynamic frames and symbols. This controller according to the specifications in [3] has six parts to perform its functions: controller host interface (CHI), protocol operation control (POC), clock synchronization process (CSP), frame and symbol process (FSP), media access control (MAC), coding and decoding (CODEC). In addition, instead of a real

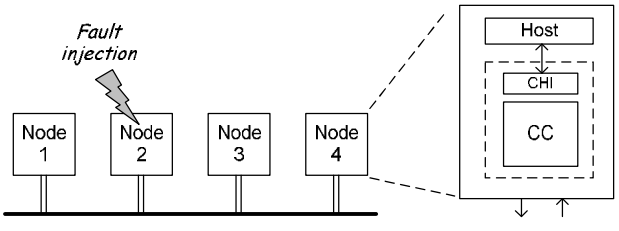

Figure 1. Experimental setup

application, a data generator is implemented to generate static frames with fixed length and dynamic frames with variable length at the start of the communication cycles.

After that, a cluster is formed consisting of 4 nodes with single bus topology. Any node is allowed to send and receive frames on communication channel. As depicted in figure 1, faults are injected in node 2 and their effects are observed in node 4. After each fault injection, the results in node 4 will be saved. Each node on this network consists of three main parts: Host that generates the frames, a controller host interface (CHI) and at lowest part there is communication controller (CC). In this experiment, faults are injected in five parts of the communication controller of the node 2, including CHI, POC, CSP, MAC and CODEC. The FSP part checks the correct timing of received frames with respect to the TDMA scheme, applies further syntactical tests to received frames, and checks the semantic correctness of received frames [3]. Thus, for the reason that the FSP part doesn't have any role in transmitting frames and error propagation to other nodes, there is no fault injection in the FSP part. The effects of fault injection are observed in communication controller of the node 4 by FSP part.

4. Experimental results

In this assessment, each experiment lasts for 3 communication cycles, in cycle 1 the faults are injected, and in cycles 1 through 3 the effects of fault injection are assessed. The results of fault injections are divided into two categories: 1) error propagation evaluation, and 2) message missing failure assessment. Following, these two categories will be discussed.

Error propagation evaluation: As discussed, the FlexRay protocol has defined three main error models that can occur in this protocol; these error models include syntax error, content error and boundary violation error. After the fault injection into node 2 in network, we investigate received errors in node 4. Table 1 contains the results of this experiment. As it shows, the fault injections in CSP part causes most content errors and boundary violation errors; the fault injections in CHI part causes most syntax errors.

Message missing failure assessment: In the first part the errors propagation in a FlexRay-based network were evaluated. These errors have potential ability to generate failures in the network. In this part, the message missing failure is assessed as the result of the fault injection. So, the messages that has been sent by node 2, is checked. This node has two slot IDs in static window and two slot IDs in dynamic window, and message transmission in dynamic window is done randomly (it can be occurred or not). As each experiment lasts 3 communication cycles, this node sends totally 9 messages during each experiment (6 messages for static window and 3 messages for dynamic window).

Table 1. Effect of fault injection in FlexRay parts

FlexRay Parts	No. of Faults	Syntax Errors		Content Errors		Boundary Violation Errors	
		#	%	#	%	#	%
CODEC	9300	457	4.91	2	0.02	164	1.76
MAC	4100	175	4.26	53	1.29	159	3.87
CSP	12480	2939	23.54	1724	13.81	2994	23.99
POC	2800	13	0.46	0	0.00	0	0.00
CHI	7000	1745	24.92	204	2.91	635	9.07
All Parts	**35680**	**5329**	**14.93**	**1983**	**5.55**	**3952**	**11.07**

For assessing this failure some counters are used. For instance, we use a counter for counting the sent messages in node 2, and a counter for counting the received messages from node 2 in node 4. So, by knowing the number of generated messages in node 2 and the number of sent messages in node 2 and the number of valid received messages in node 4, we can investigate the message missing rate in this network.

Table 2 shows the message missing rate after fault injections in the FlexRay parts. As it illustrates, the fault injections in CSP, CHI and POC lead to most message missing failures. As expected from the errors propagation results, the results of message missing failures in CSP and CHI are usual but the results of POC are unusual. In spite of its low error propagation, this part causes high message missing rate. It is because of the modes changing that occur after injecting the faults in this part. As this part controls the operation of other parts, any fault that injected in this part can change the operating mode of other parts of the node, whereas it doesn't generate errors.

In Figure 2 the message missing failures in static window are shown. Like table 2, fault injection in CSP,

Table 2. Message missing failures in FlexRay network

FlexRay part	No. of faults	No. of experiments including failure		Total messages	Missed messages	
		#	%		#	%
CODEC	9300	634	6.81	73700	3033	7.27
MAC	4100	996	24.29	36900	4046	10.96
CSP	12480	6444	51.63	112320	29412	26.19
POC	2800	1013	36.17	25200	5466	21.69
CHI	7000	3658	52.25	63000	14807	23.50
All parts	**35680**	**12745**	**35.72**	**311120**	**56764**	**18.25**

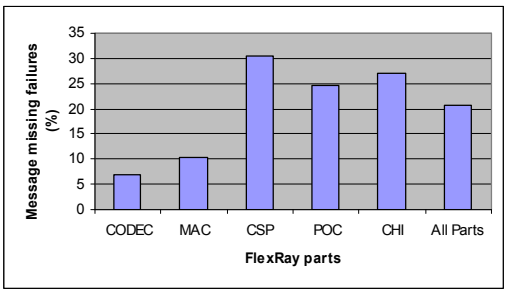

Figure 2. Message missing failures in static window

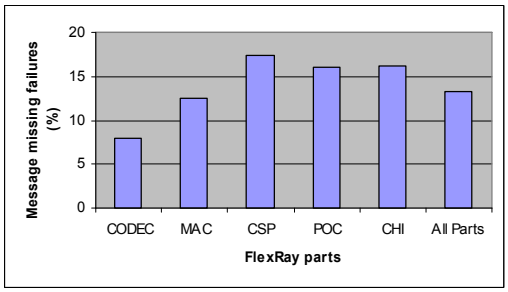

Figure 3. Message missing failures in dynamic window

CHI and POC part of the FlexRay cause most failures. The message missing failure in dynamic window is shown in Figure 3. Also in dynamic window CSP, CHI and POC generates most message missing failures but its rate is less than static window.

5. Conclusions

This paper assessed the error propagation and its effects of message missing failures in a FlexRay-based network. The assessment was based on about 35680 bit-flip fault injections inside different parts of the FlexRay communication. To do this, a FlexRay communication controller was modeled by Verilog HDL at the behavioral level. This HDL model of the controller was exploited to setup a FlexRay-based network composed of four nodes. The results of fault injection can be divided into two categories. In the first category, the percentages of faults resulting in the three kinds of errors, namely, content errors, syntax errors and boundary violation errors are characterized. Then in second category, by considering the error propagation results, the message missing failures that occur in the network were assessed. The results showed about 20% of messages that were sent in static window led to message missing failures and about 13% of messages that were sent in dynamic window led to message missing failures. The dependencies of fault locations (FlexRay parts) to this failure were also assessed. The results showed that the controller host interface and the clock synchronization process of the FlexRay were most sensitive to the message missing failures. The coding and decoding unit of the FlexRay was least sensitive to this failure.

6. References

[1] H. Kopetz, "A Comparison of CAN and TTP," Vienna University of Technology, Real-Time System Group, Research Report 23,1998.

[2] T. Pop, P. Pop, P. Eles, Z. Peng, and A. Andrei, "Timing Analysis of the FlexRay Communication Protocol," *Proc. of the 18th Euromicro Conference on Real-Time System*, July 2006, pp. 203-216.

[3] FlexRay Consortium, "FlexRay Communications System - Protocol Specification," v2.1 Revision A, December 2005.

[4] N. Navet, Y. Song, F. Simonot-Lion, and C. Wilwert, "Trends in Automotive Communication Systems," *Proc. of the IEEE*, June 2005, vol. 93, no. 6.

[5] H. Salmani, and S. G.Miremadi, "Assessment of Message Missing Failures in CAN-based Systems," *Proc. of the Parallel and Distributed Computing and Networks*, 2005, pp. 387-392.

[6] H. Salmani, and S. G. Miremadi "Contribution of Controller Area Networks Controllers to Masquerade Failures," *Proc. of the 11th Pacific Rim International Symposium on Dependable Computing*, 2005, pp. 310- 316.

[7] H. Sivencrona, P. Johannessen, M. Persson, and J. Torin, "Heavy-ion Fault Injections in the Time-triggered Communication Protocol," *Proc. of the Latin American Symposium on Dependable Computing*, 2003, pp. 69-80.

[8] H. Sivencrona, M. Persson, and J. Torin, "Using Heavy-Ion Fault Injection to Evaluate Fault Tolerance with Respect to Cluster Size in a Time-Triggered Communication Systems," *Proc. of the IEEE International Workshop on Design and Diagnostics of Electronic Circuits and Systems (DDECS-06)*, April 2003, pp. 171-176.

[9] A. Ademaj, H. Sivencrona, G. Bauer, and J. Torin, "Evaluation of Fault Handling of the Time-Triggered Architecture with Bus and Star Topology," *Proc. of the International Conference on Dependable Systems and Networks*, June 2003, pp. 123-133.

[10] R. Pallierer, M.Horauer, M. Zauner, A. Steininger, E. Armengaud, and F. Rothensteiner, "A Generic Tool for Systematic Tests in Embedded Automotive Communication Systems," *Proc. of the Embedded World Conference*, 2005.

[11] E. Armengaud, F. Rothensteiner, A. Steininger, and M. Horauer, "A Method for Bit Level Test and Diagnosis of Communication Services," *Proc. of the IEEE Workshop on Design & Diagnostics of Electronic Circuits & Systems*, 2005.

[12] FlexRay Consortium, "FlexRay Communications System - Protocol Conformance Test Specification," v2.1, December 2005.