A METHOD FOR BIT LEVEL TEST AND DIAGNOSIS OF COMMUNICATION SERVICES

E. Armengaud, F. Rothensteiner, A. Steininger
Vienna University of Technology
Embedded Computing Systems Group E182-2
Treitlstr. 3, A-1040 Vienna
{armengaud, rothensteiner, steininger}@ecs.tuwien.ac.at

M. Horauer
University of Applied Sciences
Technikum Wien
Höchstädtplatz 5, A-1200 Vienna
horauer@technikum-wien.at

Abstract. *Test and diagnosis of communication services is a pre-requisite for the validation of reliable distributed communication. In-depth observability and controllability of the bus traffic enables access to information typically filtered out by dedicated hardware. Given a fixed time interval, however, higher accuracy implies larger volumes of data that increase the test complexity and paradoxically constrain the test activity. In this paper we analyze the trade-offs between accuracy and data volume and present solutions to enable efficient test and detailed diagnosis of serial communication protocols. These trade-offs are illustrated using experiments conducted with the FlexRay protocol.*

1 Introduction

Nowadays, modern cars are forming highly complex distributed systems with up to 70 Electronic Control Units (ECU) connected to each other. Higher connectivity enables coordination between these ECUs resulting in a comprehensive view of the environment, hence, permitting efficient local processing (e.g. brake assistance). This in turn, provides the basis for dependable applications required for X-by-Wire [1]. A drawback of these systems, however, is their complexity and the difficulty to efficiently test the whole system. Although each node can be tested in isolation, the proper function of the system relies on the correct interoperation between all ECUs. Since test efforts are known to rise more than linearly with system complexity a systematic method is required providing a good test coverage. Within our STEACS project[1] particular emphasis is set on the communication network that presents the crucial part of these distributed systems.

Frequently, observation is based on some form of packet filtering using a high level view of the communication, see [2, 3] and usually implements a dedicated hardware to automatically interpret the bus traffic. Although these approaches save host processing resources, they inhibit user-specific, accurate tests of the communication medium due to the

[1] The STEACS-project received support from the Austrian "FIT-IT Embedded systems" initiative, funded by the Austrian Ministry for Traffic, Innovation and Technology (BMVIT) and managed by the Austrian Industrial Research Promotion Fund (FFF) under grant 807146. See http://embsys.technikum-wien.at/steacs.html for further information.

substantial data reduction. Bus access at the bit level is desirable in order to obtain accurate information for tests, since it enables:

1. Observability and controllability next to the physical interface of the communication controller hardware, thus, enabling “black box” test methods.
2. An accurate characterization of bit streams that is not (fully) supported by standard hardware (e.g. symbols used for signaling at the physical layer, erroneous frames, etc.).
3. Access to events that are otherwise removed at higher levels (e.g. glitches).

However, a fundamental drawback of accessing the bus traffic at the bit level is the large amount of data that needs to be transferred and processed. Furthermore, constraints in automotive embedded systems (e.g. low costs, robustness, etc.) prohibit the field-usage of COTS development tools as e.g. logic analyzers or pattern generators. Other approaches have been presented to test the lower layers of field bus systems, e.g. [4, 5]. They provide means to insert and detect a pre-defined subset of faults; however, the user has no access to the bit stream itself to enable more sophisticated processing.

2 Rationale

A fundamental problem of testing distributed communication systems is the requirement to maximize the accuracy of the tests although the data volume must be minimized. To that end, the following items need to be considered:

- Sampling frequency
- Data format (e.g. to include an identifier, a timestamp, etc.)
- Trigger and filter mechanisms

2.1 Sampling frequency

Both data accuracy and data amount are directly related to the sampling frequency. While accessing the bus at a lower abstraction level provides higher accuracy, it usually produces larger amount of data. Therefore, it is important to choose the relevant level of abstraction. To that aim, a fine grained layer model has been presented in [6] and two abstraction levels are highlighted:

- *Over-sampled stream*: This abstraction level represents the waveform at the bus transceiver interface and constitutes the highest accuracy accessible in the digital domain. It enables detection of symptoms below the bit level, e.g., glitches, line delays, jitter or bit length variation due to its continuous time scale. However, it requires a high sampling frequency.
- *Raw bit stream*: This abstraction level is the interface between the physical layer and the data link layer. Herein, the information is made up of bits with a fixed length. Thus, the use of a standard clock is acceptable, which results in a low data volume compared to the “over-sampled stream”.

The main advantage of the “raw bit stream” abstraction level is to present a concise view of the bus traffic: each bit represents a bit cell actively transmitted. However, the accuracy is limited and information below the data link layer is already abstracted. In contrast, the over-sampled stream enables a more accurate diagnosis at the cost of a data volume increased by a factor five to ten. Additionally, access to this abstraction level is mandatory for “black box” tests of the hardware where direct access to its physical interfaces is required. Figure 1 depicts an example consisting of a raw bit stream and a 6 times over-sampled stream.

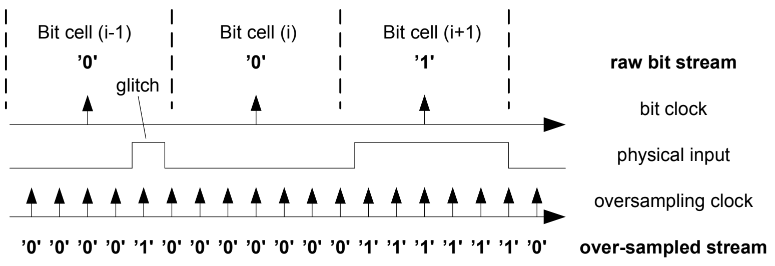

Figure 1: Example: bit stream versus over-sampled stream

2.2 Data format

The characteristic of a serial communication protocol is to transfer complex information over a single wire; the single bit information in itself becomes meaningful only in context with the whole bit stream representing the complex information. Consequently, bandwidth can be saved when grouping bits into streams. The structure overhead – mandatory for monitoring – is then limited to one tag per stream instead of one per bit.

Moreover, the sampling method applied has a direct impact on the data volume generated. *Sequential sampling* produces a data amount proportional to the acquisition accuracy but independent of the bus traffic. This method is preferable when the events to monitor are changing with a frequency close to the sample clock (e.g. raw bit stream) because no additional timing information has to be stored. *Transitional sampling* produces a data amount proportional to the bus activity but not (linearly) proportional to the acquisition accuracy. This method is more efficient when the sampling clock is much higher than the events to be monitored (e.g. over-sampled stream). If no glitches occur, one can expect the logic values of the samples to remain constant throughout the whole bit cell, and therefore produce less entries compared to the sequential sampling.

2.3 Triggering and filtering

Whereas a trigger is a mechanism to control tool activation during a predefined time window, filter (or converter) methods are based on data transformation or interpretation. Defining a trigger means defining a time window for which the tool should be activated and therefore defining two points in time respectively to start and stop the measurement. A trigger can be *continuous* when the user is accessing the entire bus traffic and the start and stop conditions are defined by the beginning and end of system operation. An *absolute* condition is defined before the experiment is started (typically according to a time base). A *relative* condition, in contrast, is derived from the experiment itself (e.g. trigger on a frame, an error, etc.). Due to the possible cyclic nature, trigger mechanisms might result to filter-like operation.

Parallel to the trigger, several filter modes can be defined. The first one is *direct access*, when the entire bus traffic (without filtering) is accessed. The second one is *transformation*, where the data format is converted to a more advantageous format or where specific (high level) information is extracted. Finally, knowledge about the correct behavior (i.e. rules the bus traffic should comply to) can be transferred to the tool to achieve *interpretation* and, hence, enable fault detection.

The usage of a filter or a trigger is not mutually exclusive. In particular, a combination of both methods frequently represents a good solution to reduce the amount of data and to obtain the required information solely.

3 A modular architecture

The aim of this section is to present the architecture developed to access the bus traffic at the bit level. In order to save effort and time we reused the existing test and diagnosis tool previously developed within the STEACS project, see [7] and added an additional queue to access the bit level information. Figure 2 presents an overview of the monitoring part; the replay part resembles a similar architecture, however, with a reversed data path. The actual implementation is based on a COTS FlexRay [8] protocol engine on the top of which different queuing units are processing different data types in parallel. As soon as a packet has been successfully computed, it is sent to a Dual Ported RAM and then to a local host for further processing. Using this modular architecture, a dedicated processing queue for the bit stream can be easily inserted.

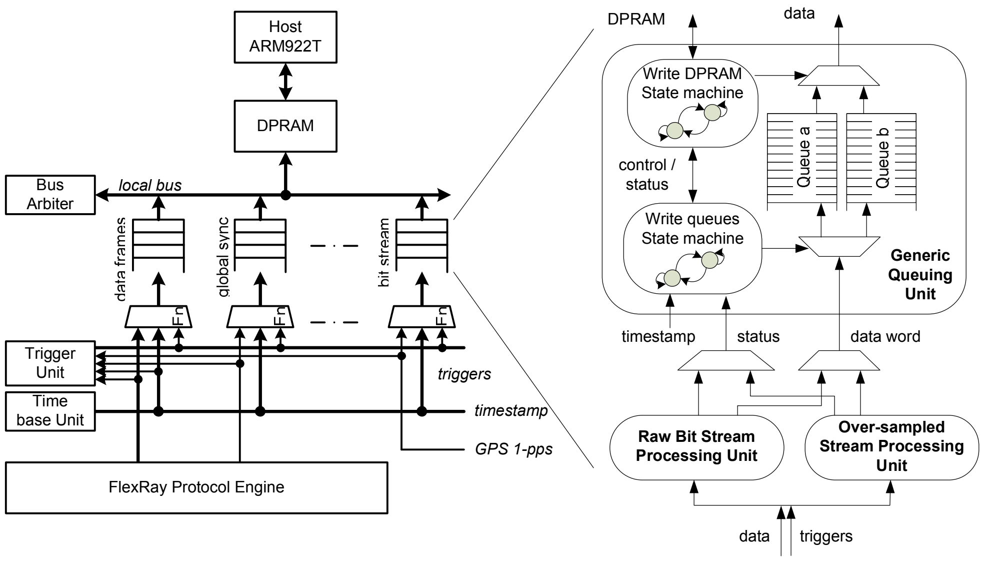

Figure 2: Global monitoring architecture of the BusDoctor

The goal of the bit stream unit is to offer a way to access the bit level both at the raw bit stream and at the over-sampled stream. Therefore, this module consists of three parts: (i) a raw bit stream and (ii) an over-sampled stream unit to process and convert the serial data to words and detect the start and stop point of the bit stream. Moreover, (iii) a generic queuing unit processes the words into packets and forwards them to the DPRAM.

The main advantages of the raw bit stream unit are its simple architecture and the lower amount of produced data. Indeed, only one bit is generated for each bit received (sequential sampling) and therefore the amount of data is comparable to the one generated when monitoring the bus traffic at the frame level (i.e., payload information without frame header or trailer). However, its accuracy is limited and information below the bit level, e.g. glitches or the bit-cell length cannot be assessed anymore. The over-sampling stream unit aims at improving this limitation and process the data directly at the physical input to provide the user with the maximal achievable accuracy. However, this is achieved at the cost of complex architecture and higher amount of produced data. Depending on the packet format, an increase of 5 to 10 must be considered when compared to the raw bit stream coding.

4 Application to FlexRay

In this section we illustrate the test and diagnosis improvements that can be achievable when monitoring and injection at the bit level is performed. For the following experiments we first recorded a fault free bus traffic, which was emanated from a FlexRay based distributed system, to a file. The bus traffic is divided into "communication cycles" that represent a repeating medium access scheme according to a configured schedule. For our experiment, a communication cycle consisted of five frames as illustrated in Figure 3.

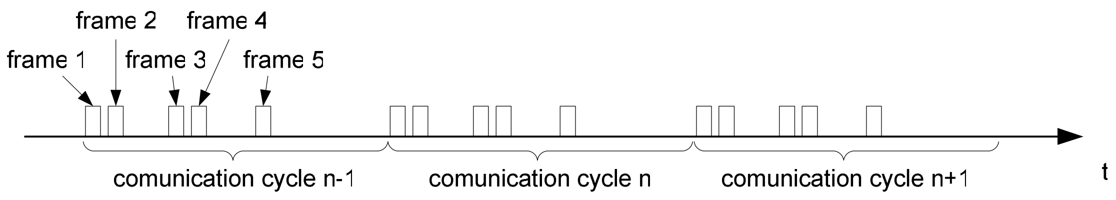

Figure 3: Bus traffic organization

Next, the file was modified to inject a fault to each tenth frame with identifier 3. The fault consisted of a single bit flip. In total, 150 frames were modified to successively alter each bit of the frame. Finally, the resulting bus traffic was replayed and monitored in parallel at the raw bit stream level for accurate diagnosis and at the frame level for comparison with a standard hardware. The results are presented in Table 1.

Table 1: Experiment results

	Number of bit flip injected	Fault detection		Fault diagnosis	
		Standard controller	BusDoctor	Standard controller	BusDoctor
Frame structure	38	87%	100%	0%	100%
Medium access	40	100%	100%	0%	100%
Frame content	72	100%	100%	0%	100%
Total	150	96%	100%	0%	100%

During this experiment, three faults location within the frame were identified. The *frame structure* represents the additional bit patterns to structure the frame (e.g. frame start/end sequence, byte start sequence). The *medium access* part provides information about the medium access correctness (e.g. cycle counter, frame identifier, CRC), and finally the *frame content* holds the transmitted payload. As expected, every bit flip was detected by our diagnosis tool while few bit flips escaped the standard controller (these bit flips split the frame into a long glitch and another correct frame tolerated by the controller). The fault diagnosis capability, however, largely differs between the two modules. While only a few flags are available for corrupted frames within the standard controller, the BusDoctor tool provides the user with the whole bit stream. Consequently, accurate tests could be carried out to identify the corrupted bit.

Access to the over-sampled stream provided us with additional information about the length of the bit cells and possible glitches. During this experiment, the bus traffic has been online interpreted to filter faulty behaviors, which has been then used to trigger bit level monitoring. This combined use of filter and trigger provided us with (i) exactly the data we were interested in, (ii) very accurate information (over-sampled stream), and (iii) enabled long observation times. Over-sampling only the faulty frames required 27 Kbytes, instead of 9500 Kbytes required to monitor the whole bus traffic with the same accuracy.

5 Conclusion

Accessing the bus traffic at the bit level enables accurate tests and allows better diagnosis of the communication services. Paradoxically, too much information might overcome the end user or impair a higher processing overhead and therefore handicap the test procedure. Consequently, the accuracy required (and therefore the abstraction level accessed) has to be carefully selected and methods to limit the data volume have to be applied. To that end, the raw bit stream and over-sampled stream abstraction levels using different sampling methods were compared. Additionally, trigger and filter mechanisms were presented to decrease the data volume while keeping the accuracy high.

A modular and structured architecture was presented to enable access (for both monitoring and injection/replay) at both the raw bit stream and over-sampled stream. The flexible implementation enables the user to efficiently switch from one view to the other according to his requirements. A prototype was integrated to the existing BUSDOCTOR, a COTS tester node from DECOMSYS. Finally, an experiment illustrated the improvements of our approach in comparison to a standard hardware. While the fault detection coverage is equivalent for both methods, diagnosis is greatly improved with the possibility to run complex algorithms directly on the faulty stream.

6 References

[1] Leen, G; Hefferman, D; "In-Vehicle Networks, Expanding Automotive Electronic Systems", IEEE Transaction on Computers, January 2002, pp. 88-93.

[2] Chandra, R; Lefever, R.M.; Joshi, K.R.; Cukier, M.; Sanders, W.H.; "A global-state-triggered fault injector for distributed system evaluation", Parallel and Distributed Systems, IEEE Transactions on, Volume: 15 , Issue: 7 , July 2004, pp. 593 - 605.

[3] Stott, D.T.; Floering, B.; Burke, D.; Kalbarczpk, Z.; Iyer, R.K.; "NFTAPE: a framework for assessing dependability in distributed systems with lightweight fault injectors", Computer Performance and Dependability Symposium, 2000. Proceedings. IEEE International , 27-29 March 2000, pp. 91 – 100.

[4] Carvalho, J.; Portugal, P.; Carvalho, A.; "A framework for dependability evaluation of PROFIBUS networks", Industrial Electronics, 2003. ISIE '03. 2003 IEEE International Symposium on, Volume: 1, 9-11 June 2003, pp. 466 – 471.

[5] Bruce, J.W.; Gray, M.A.; Follett, R.F.; "Personal digital assistant (PDA) based I2C bus analysis", Consumer Electronics, IEEE Transactions on , Volume: 49 , Issue: 4 , Nov. 2003, pp.1482 – 1487.

[6] Armengaud, E.; Steininger, A.; Horauer, M.; Pallierer, R.; "A Layer Model for the Systematic Test of Time-Triggered Automotive Communication Systems", 5th IEEE International Workshop on Factory Communication Systems, pages 275 – 283, 2004.

[7] Horauer, M.; Rothensteiner, F.; Zauner M.; Armengaud, E.; Steininger, A.; Friedl H.; Pallierer, R.; "An FPGA based SoC Design for Testing Embedded Automotive Communication Systems employing the FlexRay Protocol"; Austrochip 2004, pp. 119-125, Villach - Austria, October 2004.

[8] FlexRay Consortium, "FlexRay Communications System Protocol Specification Version 2.0", www.flexray.com, 2004.