Reliable System Design with a High Degree of Diagnostic Procedures for Embedded Systems

Michael H. Schwarz[1] **and Josef Börcsök**[2]

Department of Safety Computer Technology[1]*,*

Department of Computer Architecture and System Programming[2]

University of Kassel, Germany[1, 2]

E-mail: [1]m.schwarz@uni-kassel.de [2]j.boercsoek@uni-kassel.de

Abstract ─ Maintenance starts with reliable diagnostics. Programming Logic Controllers (PLCs) are often equipped with a high degree of diagnostic procedures in order to ensure that the processing unit is functioning correctly. It is vital to verify that the system with its programme is still within a 'healthy' state, otherwise a safety function is called and the system is brought into a safe state, or if possible, defect and malfunctioning components are exchanged during operation and the process can continue without shutting down the system. However, when it comes to smaller devices such as intelligent sensors, embedded controller devices with the functionality of an e.g. PID (Proportional-Integral-Derivative), predictive controller, filter or analytical algorithm, which is embedded into a FPGA or micro-controller then diagnostics and verification methods are often not considered in the way they should be. For example, if an intelligent sensor system is not able to diagnose that the sensor-head is malfunctioning, but the sensor-head still provides some data, then the smart algorithm bases its calculation on wrong data, which can cause a dangerous situation. This paper investigates and shows recent results to combine diagnostic methods for small scale devices. Several safety-related structures are considered with a high degree of diagnostic coverage. The paper presents relevant procedures and structures to increase the reliability of small devices without utilising a full scale microcontroller system.

Keywords ─ Reliability, safety structure, maintenance, diagnosis.

I Introduction

Reliable system design is an important and fast growing research area with a wide range of applications, especially when international standards are involved. For example, in process and chemical industries safety programming logic controllers (PLC) and safety systems are necessary equipment to protect human lives, environment and production facilities. Research and development in this section on PLC is matured, however, still on-going and far from completed but the necessary sensibility is available and present.

When it comes to sensors or actuators the same sensibility is often missing or researchers and developers are simply not aware of these issues. In an investigation by the HSE [10] where a system was subdivided into three parts as shown in Figure 1 and fatal accidents were related to the actual component that caused the accident.

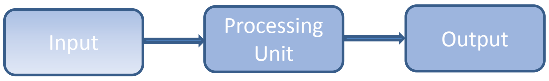

Fig. 1: System with its subdivisions

A fault in the output-section was with 50% the main cause of accidents, followed by the category of sensors with 35% and only 15% were related to the actual processing unit, as shown in Figure 2. The last part (processing unit) is the area where most of the research and development has been done in the past years.

When it comes to predictive maintenance then sensors are of great importance as they measure the health state of a system or component and either diagnose the current state immediately or send the information to a superior unit that carries out the analysis. Especially in the first case, the sensor itself is not a passive measurement device anymore but an entire system with input, processing unit and output.

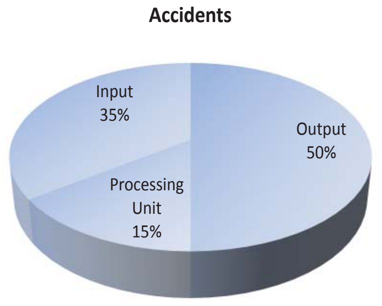

Fig. 2: Relation Accidents to causes

When a new intelligent sensor device, a new control algorithm or filter is developed, then the standard way is to use components from the shelf like a microcontroller and the algorithm is converted or directly downloaded. For standard application this is a suitable way. For more reliable design this is not enough.

The development of a reliable system starts always with a specification, where it is defined what the system should do (necessary requirements), and servers as a test bench afterwards. During the design and implementation phase the work has to be verified according to specification that the development still matches the requirements. After the development and verification, the system can installed.

When it comes to the lifecycle of a system and the causes of failures then most errors are done in the specification phase. Again this investigation has been carried out by HSE [11] and the results are shown in Figure 3.

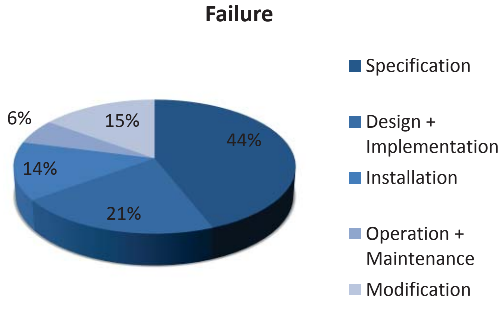

Fig. 3: Failure versus Roots

However, much work has been done, in the direction of how to get requirement formulations more precise.

The last two figures show that sensor design is important as it is the cause of many failures but the careful design from the beginning should not be underestimated. As sensors are important for maintenance and e-maintenance in order to diagnose the health state of the observed system, it is also important that the health state of the sensor itself is diagnosed, as it is vital to ensure that the right information are available to make the right judgment.

The remaining paper is structured as follows: Section 2 presents a short literature survey, section 3 describes recent results investigate within the department and section 4 draws some conclusions and state future work.

II MOTIVATION AND LITERATURE SURVEY

Different aspects of the development of safety and reliable systems are detailed in the publication by Schwarz [18], in this case the development of a sensor system was described using Matlab/Simulink. Two main approaches exist; the first uses an automated generated c-code which is executed by an operating system, the second uses a VHDL approach directly for a hardware board. However, the development follows in both cases the v-model [12] methodology for the design of reliable / safe systems. The system can be tested during every stage of each design phase. Different architectures are detailed as well different tests and checks.

Marek Sniezek and Josef von Stackelberg [19] describe in their publication on *'A fail safe programmable logic controller'* a hardware approach to develop a safe controller that fulfills the requirement of the international standard IEC 61508 [12] for the safety integrity level 3(SIL 3). They describe a novel safe comparator strategy to distinguish between different safety faults and to achieve a fast reaction time.

Riccardo Mariani and Gabriele Boschi [16] are dealing with robust memory approaches. International standards such as the IEC 61508 [12] or others demand that developer considers errors and faults in the memory of embedded controller systems and deploy methods and techniques to deal with those effects.

Grießing et al. and Alvarez et al from the University of Vigo, Spain present another approach direction. Grießing et al. from the University of Graz and associated companies [7], [8], [9] describe a development of a safety related function using Complex Programming Logic Devices (CPLD). It is stated by the authors that the development follows safety related standards such as the IEC 61508 [12] and the ISO 13849 [13]. The papers describe the development, several testing procedures and implementation using the derived system to guard a power drive system. Alvarez et al. [2] used PLDs to implement a 2oo3D system, which consists of a 2oo3 safety architecture with an additional diagnostic system. They claim that their approach meets the required safety performance but is more flexible and cheaper as a full micro-processor system.

Rapid prototyping and a full development suite from simulation to hardware design are an increasing research topic. Reyneri [17] describes an interesting code design system for rapid prototyping using FPGA systems. The described system contains Mathworks tools for high-level description languages and a simulation environment. The user can simulate and optimise system and architectural parameters before it is downloaded onto a user-defined FPGA.

Krakora and Hanzalek [14] from the Technical University in Prague present in their publication a testing methodology for hybrid *Hardware-in-the-Loop* tests, for discrete events, time automata continuous systems and differential equations utilising FPGA technology. Their implementation concentrates on a discrete event system linked with continuous systems implemented as filters using fixed-point arithmetic. They use Matlab/UPPAAL in combination with FPGA based testing tool.

Alberto et al. [1] describes an innovating filtering structure to detect gas particles using a FPGA system for processing the data and signals. Different filter structures were tested to achieve a high working frequency.

The research by Astarloa et al. from the University of the Basque County [3] includes the development of a PID controller IP core to transfer computational expensive parts into hardware on an FPGA. This system is self-reconfigurable and different subsystems with altered features can be loaded and started during run time.

Elhadef et al. [5] from the University Ottawa proposed a self-diagnostic technique exploiting generalised comparison models to detect several faults. They used an artificial immune system in order to carry out the diagnosis. Another attempt uses a multi-layered neural network [6] considering permanent faults in a t-diagnosable system.

Another interesting approach is presented by Machado et al. [15] where simulation and formal verifications are combined to develop a reliable and safe controller. Timed Automata formalism and UPPAAL real time model checker are used to validate the derived model.

III Design

The proposed structure and system has been developed with Matlab / Simulink, where it can be tested and simulated, before it is transferred into VHDL code and downloaded onto an FPGA [4]. Another possibility would be to develop only the application [16] and embed the developed software in an operating system and use a 1oo2 safety processor system. The ideas here are to derive, simulate and test a system suitable for small applications and after the results are satisfying to implement as much functionality as possible into hardware rather in software, but without losing any reliability feature.

a) Hardware

Figure 4 shows a schematic of a normally used system, which is classified as a 1oo1 structure as described before.

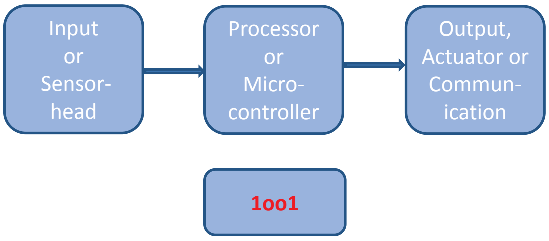
Input or Sensor-head
Processor or Micro-controller
Output, Actuator or Commun-ication
1oo1

Fig. 4: Sensor System

For standard applications or demonstration purposes such a system might be sufficient but it contains no reliability and safety enhancements at all. If this system gets stuck or the execution freezes then it is not possible to call a safety function and to close the process under control or if this architecture is used in a sensor system, then the system cannot inform the higher processing unit e.g. a PLC (Programming Logic Controller) about the problem and to close its functionality in a systematic manner.

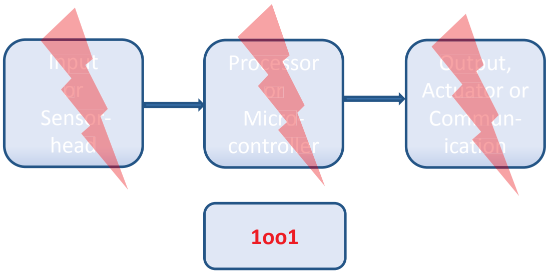
Input or Sensor-head
Processor or Micro-controller
Output, Actuator or Commun-ication
1oo1

Fig. 5: Sensor System with an error

Figure 5 illustrates the problem, that when a part fails the information cannot be delivered to the next processing hierarchy.

The only safety function would be a watchdog which reacts with a reset of the entire system, when the watch dog time is elapsed. A system reset during a production can cause unpredictable situations and can harm people nearby and can cause hazardous situations. A safety function could be that the system creates an alarm or an event and send a message to a superior system or monitoring system and informs it that the particular system has to be shut down, necessary valves for examples are closed and relevant data is stored to analyse it afterwards or whatever functionality is considered for the safety function.

b) Reliable Hardware

In order to increase functional safety and availability of a system a multi-processor architecture in an *N* out of *M* structure is often recommended. A 1oo1 architecture as shown in Figure 4 is the simplest and often used system, but contains no safety architecture at all. If one of the subsystems (Input Processing Unit, Output) fails then the entire system might fail and a safety function might not be initiated to bring the system into a safe state.

A 1oo2 system, as shown in Figure 7, possesses two independent paths to call the safety function, if one of the two systems fail then the other one is still able to call the safety function. A schematic is shown below.

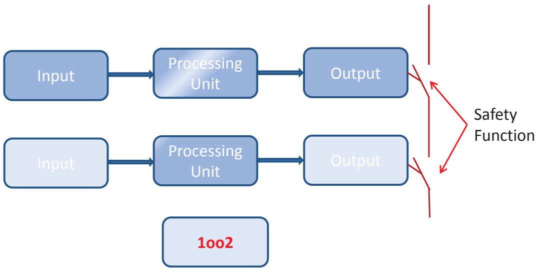

Fig. 6: 1oo2 Safety System

A 1oo2D system is similar compared to a 1oo2 system, but it possesses a high degree of diagnostic procedures. The system is self-tested after each cycle and if a difference is diagnosed then the system is brought into a safe state. High diagnostic procedures are necessary to verify that each processor is functioning correctly.

A system with a higher availability such as a 2oo2 has a redundant structure but a lower reliability as both subsystems have to call the safe function. A schematic is shown in Figure 8.

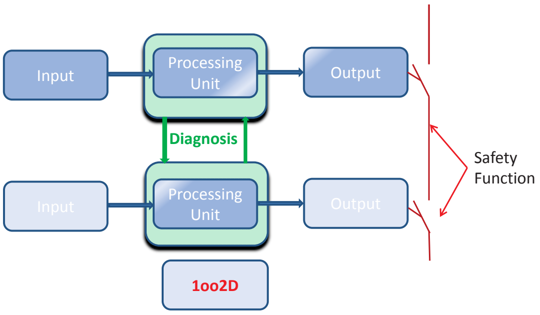

Fig. 7: 1oo2D Safety System

A 2oo2 system is actually not a safety system as both systems have to call the safety function to bring the system into the safe state. However, a combination of a system with a higher degree of safety and availability lead to systems which can tolerate faults and can continue without shutting down the system. Many processes cannot be stopped but have to be continued until the process is completed, but such systems are not considered in this paper.

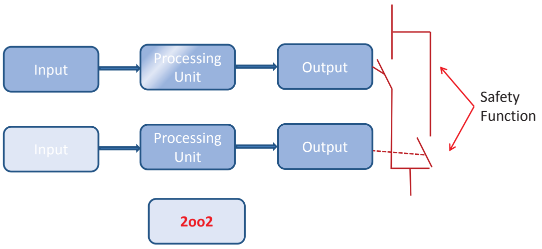

Fig. 8: 2oo2 Safety System

When developing reliable sensor systems, controller systems or filters, then using full multi-core architectures might become oversized in terms of its functionality not architecture, to execute few relatively simple mathematical operations, which are always carried out in the same way. However, sensor, controller or filter systems are becoming more reliable and are equipped with more intelligent methods and procedures. Sensors are not simple devices anymore to measure a physical value, but becoming complex computer systems by their own, e.g. equipped with Ethernet communication, wireless communication, analysis methods, filters; the development of such peripheral systems should follow the same development procedures and guidelines as safety and reliable systems.

c) Reliable Design

The structure considered in this paper is shown in Figure 9. A 1oo2 architecture should be used, which uses two different inputs, two processing units and two independent outputs. Both processing units possess diagnostic methods and the results are compared with each other. Diagnostic procedures are able to call the safety function and to bring the system into a safe state. Also, the processing unit itself has diagnostic procedures (in the application itself), which can detect that the information itself is not correct and either correcting procedures are called or if this is not sufficient enough, the safety function will be called as well.

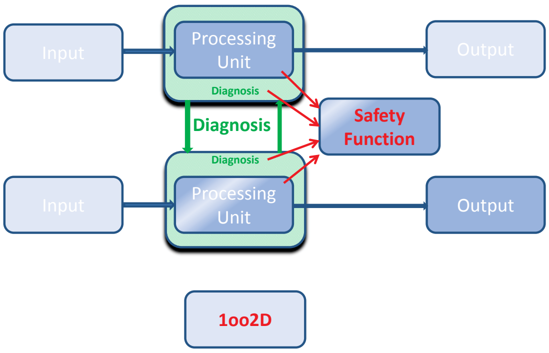

Fig. 9: 1oo2D Sensor System

The safety function could be the transmission of an error code to the superior unit, e.g. a PLC which processes further the received value. This structure does not use a majority or minority voting, which can be found in some applications.

d) Software application structure

The figure below shows the normal application structure of a sensor system. Some diagnostic procedures are also implemented in this layer. But these diagnostics are concerned of the integrity of the processing value and not with the underlying diagnostics to determine the integrity of hardware and the overall system.

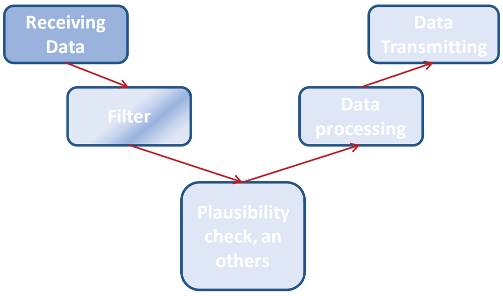

Fig. 10: Sensor Application Structure

The received data is firstly filtered due to noise, afterwards the data is verified that the value is within the allowed ranges otherwise the safety function is called. Afterwards, the data is processed and analysed which is the normal operation of the system and finally the data is sent to a superior unit, for example a PLC to be further processed.

The plausibility check is one test of others to ensure that the received process value is correct, as shown below. Other tests would be to determine the gradient and to detect a steep increase which can be an indication of an error.

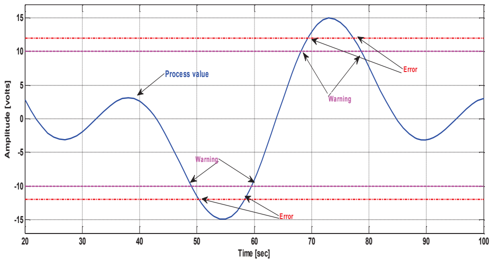

Fig. 11: Test procedures at application layer

The plausibility check mimics test procedures of industrial systems. The standard input value should be within the range of e.g. ± 10volts. The inputs are developed in such a way that they are able to measure inputs above the standard range (which is also a protection for the inputs), but provide a warning if the process value is above or below the ±10volts. If the value approaches the ±15 then an error message is provided. In the same way this plausibility check works.

The next diagnostic procedure is not situated in the application layer, where the value is tested but functionality of software should be tested, that it is still functioning tin a correct way, especially when microcontrollers (without an operating system) or FPGA systems are considered. The test procedure is explained using a filter as shown below, but it is valid for the plausibility checks and for the data processing algorithm as well, because every method has to be tested and diagnosed as well.

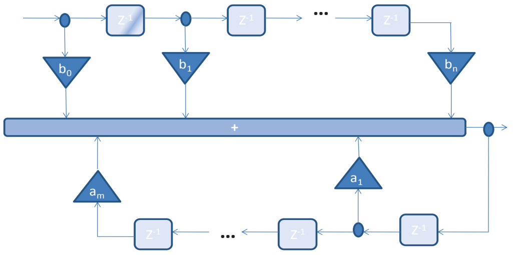

Fig. 12: IIR - Filter

The filter consists of memory blocks, summation and multiplication and this has to be tested, that the filter itself still has this functionality. Figure 13 illustrates the filter under normal conditions.

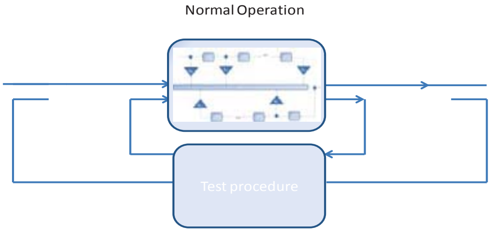

Fig. 13: normal operating mode

During the test phase the IIR-Filter is tested. Firstly, all values stored in the memory have to be stored in a separate memory block. Every value has to be read twice, to ensure that the reading was done correctly. Afterwards, the filter is prepared with predefined values and a defined sequence is written to the inputs and the filtering procedure is carried out. Afterwards the results are compared with expected values. If a difference is identified then the filter is not working correctly and system has to be brought into the safe state. Figure 14 illustrates the procedure.

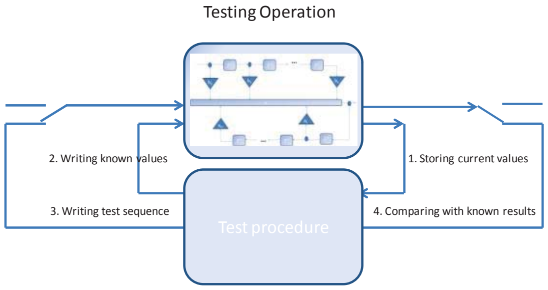

Fig. 14: Testing mode

If the test was successful, then the filter has to be loaded with the original values. It has to be ensured that the filter values are correct, therefore, the values are read back again and compared. This is demonstrated in Figure 15.

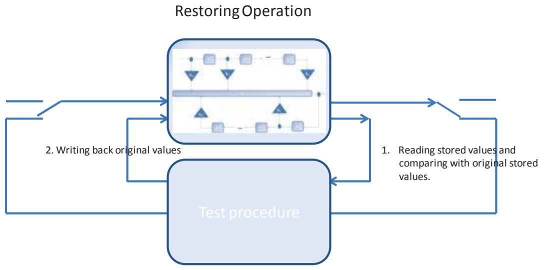
Restoring Operation
2. Writing back original values
1. Reading stored values and comparing with original stored values.
Test procedure

Fig. 15: Restoring mode

During normal operation, the test procedure has to be validated, then also this one can be damaged or alternated. This test has to be carried out in regular intervals to ensure that system is functioning correctly.

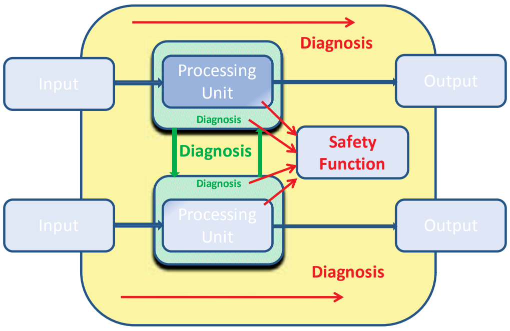

Fig. 16: System test

All the other functions have to be tested and validated in a similar way to ensure that overall function is in good condition. The diagnostics has to be done in both processors and the results have to be compared. Additionally to the processor, the peripheral components input and outputs have to be validated as well as demonstrated in Figure 16.

IV Conclusions

Reliable system design and diagnostic is an important issue not only for large systems as programming logic controllers but also for sensors and actuators. It is not only necessary to maintain and indentify the health state of a plant, motor or compressor but also to indentify the health state of the sensors and actuators itself. This paper presented a strategy to develop a sensor system in a more reliable structure. This starts with the selection of an appropriate architecture via various test and diagnostic procedures.

References

[1] Alberto D., Falletti E., Ferrero L., Garello R., Greco M. Maggiora M., 2009. FPGA implementation of digital filters for nuclear detections. *Nuclear Instruments and Methods in Physics Research A*, 661, pp. 99-104.

[2] Alvarez J., Marcos J., Fernandez S. 2005. Safe PLD-based programmable controllers. In the proceedings of International *Conference on Field Programmable Logic and Applications*. Tampere, Finland pp. 559-562

[3] Astarloa A., Lazaro J., Bidarte U., Jimenez J., Zuloaga A., 2009. *FPGA technology for multi-axis control systems*. Mechatronics 19, pp. 258-268.

[4] Burunsus C. 2011. Entwurf von zuverlässigen Simulink-Diagnose-Strukturen für den Einsatz auf FPGA Systemen (Design of reliable Simulink-diagnostic-structures for the use in FPGA systems) *Master-Thesis, 2011, University of Kassel*.

[5] Elhadef M., das S.,Nayak A., 2006. A Novel Artificial-Immune-Based Approach for System-Level Fault Diagnosis. *Proceedings of the First International Conference on Availability, Reliability and Security (ARES'06)*. Vienna, Austria, pp. 166-173.

[6] Elhadef M., Nayek A., 2010. A Novel Generalised-Comparison-Based Self-Diagnosis Algorithm for Multiprocessor and Multicomputer Systems using a Multilayered Neural Network. *In the proceedings 13th IEEE International Conference on Computational Science and Engineering*. pp 245-252.

[7] Grießing G., Mader R., Seger C., Weiß R., (2009). Fault Insertion Testing of a Novel CPLD-based Fail-Safe System. *In the Proceedings of Design, Automation & Test in Europe Conference & Exhibition*, 2009. DATE '09, pp 214-219

[8] Grießing G., Mader R., Seger C., Weiß R., (2010a). A CPLD-based Safety Concept for Industrial Applications. *In the Proceedings IEEE International Symposium on Industrial Electronics (ISIE)*, Bari, Italy, pp. 3027 - 3032

[9] Grießing G., Mader R., Seger C., Weiß R., (2010b). Design and Implementation of Safety Functions on a Novel CPLD-based Fail-Safe System Architecture. *In the Proceedings of 17th IEEE International Conference and Workshops on Engineering of Computer-Based Systems*. Oxford, United Kingdom. pp. 206-212.

[10] Health & Safety Executive (HSE) UK, 1996. The setting of safety standards: *A report by an interdepartmental group of external advisors.* London: HM stationery office, United Kingdom.

[11] Health & Safety Executive (HSE) UK, 1995. Programmable electronic systems in safety-related applications, part I. London: HM stationery office, United Kingdom.

[12] IEC 61508, 2000. International Standard 61508 Functional Safety of Electrical/Electronic/ Programmable Electronic Safety Related Systems. International Electrochemical Commission.

[13] ISO 13849, 2006. Safety of machinery -- Safety-related parts of control systems -- Part 1: General principles for design. International Organization for Standardization.

[14] Krakora J., Hanzalek Z., 2008. FPGA based tester tool for hybrid real-time systems. *Microprocessors and Microsystems* 32, pp. 447-459.

[15] Machado J., Seabra E., Campos J.C, Soares F., Leão C. P, 2011, Safe controllers design for industrial automation systems. *Computers & Industrial Engineering 60* (2011) pp. 635–653

[16] Mariani R., Boschi G. 2005, A system-level approach for embedded memory robustness. Solid-State Electronics 49 (2005) pp. 1791–1798

[17] Reyneri L.M., 2004. A Simulink-based hybrid codesign tool for rapid prototyping of FPGA's in signal processing systems. *Microprocessor and Microsystems* 28, pp. 273-289

[18] Schwarz M. H., Sheng H., Batchuluun B., Sheleh A. Chaaban W. Börcsök J. 2009. Reliable Software Development Methodology for Safety Related Applications - From Simulation to Reliable Source Code. XXII *International Symposium on Information, Communication and Automation Technologies*, Sarajevo, Bosnian-Herzegovina.

[19] Sniezek M. von Stackelberg J. 2003, A fail safe programmable logic controller. *Annual Reviews in Control* 27 (2003) pp.63–72