International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:7, No:7, 2013

Automotive ECU Design with Functional Safety for Electro-Mechanical Actuator Systems

Kyung-Jung Lee, Young-Hun Ki, and Hyun-Sik Ahn

***Abstract*—**In this paper, we propose a hardware and software design method for automotive Electronic Control Units (ECU) considering the functional safety. The proposed ECU is considered for the application to Electro-Mechanical Actuator systems and the validity of the design method is shown by the application to the Electro-Mechanical Brake (EMB) control system which is used as a brake actuator in Brake-By-Wire (BBW) systems. The importance of a functional safety-based design approach to EMB ECU design has been emphasized because of its safety-critical functions, which are executed with the aid of many electric actuators, sensors, and application software. Based on hazard analysis and risk assessment according to ISO26262, the EMB system should be ASIL-D-compliant, the highest ASIL level. To this end, an external signature watchdog and an Infineon 32-bit microcontroller TriCore are used to reduce risks considering common-cause hardware failure. Moreover, a software design method is introduced for implementing functional safety-oriented monitoring functions based on an asymmetric dual core architecture considering redundancy and diversity. The validity of the proposed ECU design approach is verified by using the EMB Hardware-In-the-Loop (HILS) system, which consists of the EMB assembly, actuator ECU, a host PC, and a few debugging devices. Furthermore, it is shown that the existing sensor fault tolerant control system can be used more effectively for mitigating the effects of hardware and software faults by applying the proposed ECU design method.

***Keywords*—**BBW (Brake-By-wire), EMB (Electro-Mechanical Brake), Functional Safety, ISO26262.

I. Introduction

THE number of Electro-Mechanical actuators in a vehicle is continuously increasing for various purposes in chassis systems and body systems of a vehicle. As an example, the AC motor is used as the EMB actuator to make braking torque while being positioned at each wheel. It operates front/rear wheel electric brakes by calculating the required brake forces in ECU via signal of the electric pedal which identifies the driver's intention for braking. The basic configuration of a BBW system consists of a pedal simulator, a main controller, electro-wedge brakes (EWB) and electro-mechanical brakes (EMB). EMB and EWB are placed at each wheel with optional independent actuator controllers. This type of brake system can improve significantly the braking performance compared with conventional hydraulic brake system, resulting in great benefits on energy consumption, weight reduction, noise and space packaging. Vehicle safety and ergonomics could also be improved by the added electronic control systems. However, as there is no mechanical linkage between command control device and actuation device, a fault could occur in any component of the BBW system resulting in an unwanted critical accident if a fault is not detected in real time. As the core conducts system application algorithms, the controller is safety-critical in an EMB system: a random hardware error could result in false command or delay response time, leading to unpredictable hazardous events. These potential risks necessitate a safety system development process for microcontroller with compliance to ISO26262 standards [1], [2].

IEC61508 has been the dominant international standard of functional safety for Electrical/Electronic/Programmable Electronic Safety-related Systems. However, originated by process and automation industries, the application of IEC 61508 in automotive industry reveals several critical problems, since it is not adapted to real-time systems or to automotive development life cycles. As a result, the ISO technical committee has been working on ISO26262, in the aim of creating a domain specific standard for E/E Systems inroad vehicles. This standard provides an automotive safety lifecycle that encompasses principal safety activities during the concept phase, product development and product release and adopts a customer risk-based approach for determining risk classes at vehicle level. It uses an Automotive Safety Integrity Level (ASIL) for specifying qualitative and quantitative requirements for safety related functions to be implemented by E/E systems and provides ASIL-dependent requirements for the whole lifecycle of E/E system(including hardware and software product development), as well as for confirmation methods and measures to ensure sufficient safety [3].

In the automotive industry, a microcontroller and software components implementing ECU has been developed and treated as one of safety element out of context (SEooC) i.e. generic element which is not developed for a specific item. The generic SEooC can be integrated into a specific item with assumptions on safety requirements which are allocated to the element by higher design level, i.e. system-level design and on the design external to the element. Examples of assumptions on system-level design could be listed as below [4].

a) The system will implement safety mechanism of detecting and monitoring the power supply of the microcontroller (MCU) to detect over voltage and under voltage failure modes.
b) The system will implement window watchdog safety mechanism to the MCU to detect either clocking or program sequence failures of the MCU.

Kyung-Jung Lee and Young-Hun Ki are graduate students of the Department of Electronics Engineering, Kookmin University, Seoul, Korea.
Hyun-Sik Ahn is the Professor of Department of Electronics Engineering, Kookmin University, Seoul, Korea (corresponding author to provide phone: +82-2-910-4709; fax: +82-2-910-4449; e-mail: ahs@kookmin.ac.kr).

c) A software test to detect latent faults in safety mechanism implemented by HW in the MCU.

In this paper, we propose an approach of how to integrate microcontroller as a SEooC into ECU design and what safety measures has to be considered to be compliant with required ASIL level, based on microcontroller with asymmetric dual-core architecture and external signature watchdog. It is also shown that the experimental result of proposed approach using EMB HILS. For ECU design, Infineon's 32-bit microcontroller TriCore and an additional external watchdog device are used

II. Hazard Analysis and Risk Assessment of EMB System

After safety goals are defined based on a preliminary functional architecture, hazard analysis and risk assessment can be carried out, as previously mentioned. This involves the categorization of ASIL standards at the vehicle and sub-system levels. The ASIL categories are severity (S~S3), exposure (E1~E4), and controllability (C0~C3), and the guidelines for rating each of these categories are listed in Table I. Then, as given in Table II, ASIL levels are assigned based on the S, E, and C ratings. As can be seen from this table, ASIL is categorized into four levels, A, B, C, and D, where A indicates the lowest risk and D, the highest. It should be noted that QM in the table denotes quality management and implies that normal quality management is sufficient for the target element to be safe and that no extra functional safety design considerations are required [5]-[8].

TABLE I
Description of Severity, Exposure, and Controllability Ratings

Severity		Exposure		Controllability	
S0	No injuries	**E1**	Very low probability	**C0**	Controllable in general
S1	Light and moderate injuries	**E2**	Low probability (<1%)	**C1**	Simply controllable (>99%)
S2	Severe injuries (survival probable)	**E3**	Medium probability (1%~10%)	**C2**	Normally controllable (>90% of drivers)
S3	Life-threatening injuries	**E4**	High Probability (>10%)	**C3**	Difficult to control (<90% of drivers)

The first step in risk assessment is to prescribe ASIL levels for the vehicle under various driving scenarios. For a failure in the braking system under normal vehicle-level driving conditions, the severity and controllability levels are usually assumed as S3 and C3, respectively. In this scenario, the braking system is considered as a black box, and it is assumed that any fault is a total fault in the braking function. The difference in rankings at the vehicle level occurs because of varying levels of exposure. For example, driving in the city, driving on country roads, driving on highways, and conducting parking maneuvers were all assumed as E4 cases in this study. In comparison, other driving situations such as driving through a tunnel, driving on ice or reversing, and vehicle being towed were assumed as E3, E2, and El, respectively. Table III summarizes the ASILs for a few vehicle-level driving scenarios, and shows the influence of exposure (E) on the assigned ASIL.

TABLE II
ASIL Levels According to S, E, and C Ratings

		Controllability		
Severity	**Exposure**	C1	C2	C3
S1	E1	QM	QM	QM
	E2	QM	QM	QM
	E3	QM	QM	ASIL A
	E4	QM	ASIL A	ASIL B
S2	E1	QM	QM	QM
	E2	QM	QM	ASIL A
	E3	QM	ASIL A	ASIL B
	E4	ASIL A	ASIL B	ASIL C
S3	E1	QM	QM	ASIL A
	E2	QM	ASIL A	ASIL B
	E3	ASIL A	ASIL B	ASIL C
	E4	ASIL B	ASIL C	ASIL D

TABLE III
Vehicle-Level Asils Assigned for Various Driving Scenarios

Level	Req.	Fall Mode	Driving Scenario	Failure Impact	S	E	C	ASIL	Safety Goal
Vehicle	Brake Request	No Braking	City	Vehicle Cannot Stop	S3	E4	C3	**D**	Braking
			Highway		S3	E4	C3	**D**	
			Tunnel		S3	E4	C3	**C**	
			Ice Road		S3	E4	C3	**B**	

III. Functional Safety Compliant Hardware Architecture

According to ISO26262, faults in the safety-related Hardware elements of the E/E system, i.e., SPF (Single-Point Fault) or RF (Residual Fault), which can potentially to lead to a safety-critical malfunction, and LFs (Latent dual-point Faults) that remain undetected within a prescribed time interval, should be evaluated for Hardware architectural metrics. The metrics for the failure rate associated with each of the abovementioned Hardware faults, i.e., SPFM (SPF Metric) and LFM (LF Metric), are evaluated as follows [9].

$$SPFM = 1 - \frac{\sum (\lambda_{SPF} + \lambda_{RF})}{\sum \lambda} \quad (1)$$

$$LFM = 1 - \frac{\sum (\lambda_{LF})}{\sum (\lambda - \lambda_{SPF} - \lambda_{RF})} \quad (2)$$

where λ_{SPF} is the failure rate associated with SPF, λ_{RF} is the failure rate associated with RF, λ_{LF} is the failure rate associated with LF, λ_S is the failure rate associated with SF(safe faults)

$$\lambda = \lambda_{SPF} + \lambda_{RF} + \lambda_{MPF} + \lambda_S$$

For the required ASIL level, Clause 8 of ISO26262-5:2011 provides the quantitative reference target values for SPFM and LFM, as summarized in Table IV. For instance, the SPFM and LFM should be equal to or higher than 99% and 90%, respectively, for achieving ASIL D.

TABLE IV
SAFETY ARCHITECTURE METRIC TARGET VALUE

	ASIL B	ASIL C	ASIL D
SPFM	≥90 %	≥97 %	≥99 %
LFM	≥60 %	≥80 %	≥90 %

For achieving the SPFM and LFM prescribed under a target ASIL level, several safety-integrity functions and safety mechanisms against SPF and LF are required. DC (diagnostic coverage), or the proportion of hardware element failure detected or controlled by the implemented safety mechanisms, should be evaluated to generate a rationale for compliance with the required SPFM and LFM levels and choosing an appropriate safety mechanism to detect element failure.

Generally, an E/E system consists of hardware elements, power supply, processing unit, memory, digital I/O, RAM, sensors, actuators, etc. ISO26262-5:2011, Annex D, summarizes the typical faults associated with each hardware element and the guidelines for DC. The following Table V lists such examples for processing units.

TABLE V
TYPICAL FAULTS OF HARDWARE ELEMENTS AND DIAGNOSTIC COVERAGE

Element	DC for failure modes		
	Low (60%)	Medium (90%)	High (99%)
Control logic	No code execution Execution too slow Stack overflow/underflow	Wrong coding or no execution Execution too slow Stack overflow/underflow	Wrong coding, wrong or no execution Execution out of order Execution too fast or too slow Stack overflow/underflow

TABLE VI
SAFETY MECHANISM FOR ELEMENTS FAULTS

Safety mechanism/measure	Typical DC considered achievable
Self-test by software: limited number of patterns (one channel)	Medium
Software diversified redundancy (one hardware channel)	High
Reciprocal comparison by software	High
HW redundancy (e.g. Dual Core Lockstep, asymmetric redundancy, coded processing)	High

Additionally, the standard lists the typical safety mechanisms associated with these element faults and categorizes the effectiveness of these safety mechanisms by ranking the DC as low, medium, or high corresponding to typical coverage levels at 60%, 90%, and 99%, respectively. Such examples for processing units are shown in Table VI.

For compliance with ASIL-D based on the abovementioned guidelines, a safety mechanism with high DC level (99%) for processing units could be implemented. For example, self-test by software in one hardware channel, i.e., processing unit, may only achieve medium-level DC (90%), thus resulting in an ASIL-B-compliant system. For details on the safety mechanism implementation methods for achieving the required DC level, refer to ISO26262-5:2011, Annex D.

Another important factor that should be considered for ensuring the achievement of a functional-safety-oriented design is the reduction or mitigation of the risk arising from common-cause failure, i.e., simultaneous failure of two or more elements of an item.

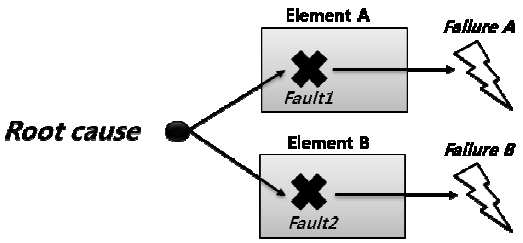

Fig. 1 Common-cause failure

The DC of the safety mechanism implemented in an item could be improved by avoiding common-cause failure by providing diversity, i.e., different solutions satisfying the same requirement with the aim of independence, in software and hardware implementation.

Diversity can be implemented, for example, using independent hardware resources, e.g. processing unit, memory, etc., in monitoring paths that are different from the primary path.

Although the diversity and redundancy of the safety measures are implemented in a primary as well as redundant/monitoring path within one processing unit, an additional external watchdog processor with an independent clock source and power supply should be used for verifying primary processing unit operation for avoiding the risks associated with potential common-cause failure. For example, the safety mechanism in one processing unit may not be executed at all if common-cause failure occurs at either the power supply or the clock source, which are shared by the primary and monitoring/redundant paths.

Additionally, ISO26262-5:2011, Annex D, lists the guidelines for ensuring that typical safety mechanisms achieve the required DC level.

Based on the above guidelines, a simple timed-window (temporal) watchdog or signature (logical) watchdog by diagnosis via question and answer without the timed-window function may only achieve a medium DC level. Thus, the combination of a temporal and logical monitoring watchdog should be used to achieve a high DC level compliant with ASIL-D.

A. Asymmetric Dual-Core Microcontroller Architecture with External Signature Watchdog

Infineon's 32-bit microcontroller provides the ideal hardware architecture for safety integrity systems. It incorporates a high-performance primary processing unit called TriCore and a separate secondary processing unit with different architecture called PCP (peripheral controller processor), thus providing the asymmetric hardware redundancy mentioned in Table III.

Notably, each of these processors has its own independent instruction set for reducing the risk resulting from systematic failure or common-cause failure by using the same hardware architecture and software development tools, e.g., the compiler

between the two processing units.

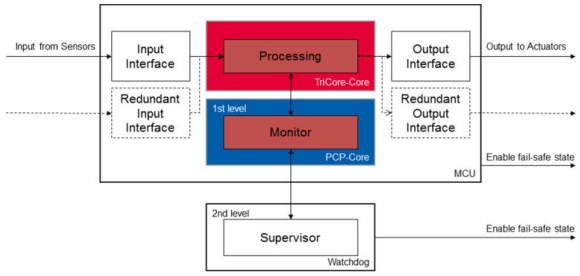

Fig. 2 Typical use case of asymmetric dual-core architecture with external watchdog

In addition to this asymmetric dual-core microcontroller, Infineon's CIC61508 is used as a second-level external signature watchdog or supervisor. At the ECU design stage, this external watchdog should be implemented with clock source and power supply that are independent of the primary microcontroller. The CIC61508 provides power supply monitoring for the primary microcontroller and the combination of temporal and logical monitoring functions via question and answer on the serial communication line. Additionally, it could offer a path for driving the system to a safe state in the event that safety-critical failure is detected in the primary microcontroller unit. Fig. 2 shows a typical use case with the TriCore microcontroller and CIC61508 [10].

The safety driver uses the TriCore to provide a reliable, self-testing computing environment for safety-critical and high-integrity applications. The main components of the safety driver include three monitor programs, which run within the PCP. Each of these monitors is used for testing a particular mode of operation of the application code running on the main TriCore processor. The safety monitor programs run a series of cyclic validation tests to ensure accurate operation of both the TriCore hardware and the application software. Additionally, the safety driver provides on-demand tests, which are invoked directly by the application software to validate its operation. The architecture of the external safety monitor is shown in Fig. 3.

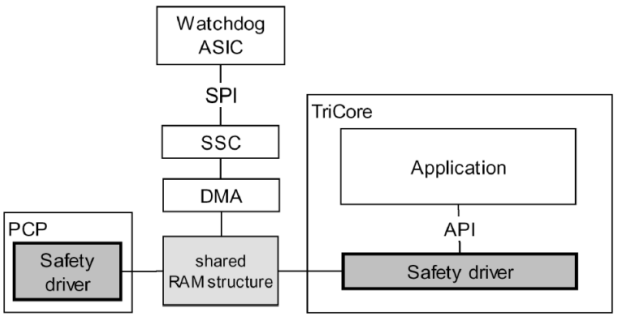

Fig. 3 External safety monitor

IV. Functional Safety Compliant Software Implementation

The safety integration layer provides an API (Application Programming Interface) that handles all necessary communication to and from the safety driver monitor programs. The safety driver monitor consists of four monitor programs (running in the PCP): the supply voltage monitor, sequencer, system monitor, and task monitor. Once the main application has started, an advanced external safety monitor that acts as a system watchdog further verifies PCP operation accuracy. These features are provided by Infineon's CIC61508 safety monitor device or Bosch's CY32x watchdog ASIC. The safety driver contains a dedicated handler for the external safety monitor; this runs within the PCP. If this handler fails to service the external safety monitor in the correct manner, the monitor times out and disables the safety path.

A. Supply Voltage Monitor

The CIC61508 can monitor up to four voltages, sampled at every heartbeat. Typically, these voltages are the power supplies to the host CPU or other safety-critical hardware in the system. The user can program the range of each voltage using the NVM. Voltage sampling is initiated upon a reset of the CIC61508. The sampled voltages are updated in the respective SFRs, and the host can read these voltages using the coherent read mechanism.

Voltage sampling can be suspended for one heartbeat by invoking the voltage injection feature. The voltage count value should be provided, instead, by a software write onto the voltage monitor registers for that channel. The voltage threshold test is carried out as before but is based on this software-written value. This can be used to deliberately inject incorrect voltage readings to verify whether the pass counter system accurately detects voltage errors.

In all cases, the pass counter of the voltage monitor is incremented if the result is valid (voltage is in the range), or decremented if the result is invalid (voltage is outside the range).

B. Sequencer

The sequencer tests the series of answers generated by the host controller at regular time intervals. It then updates the request number (question) and expects the host to send the relevant answer. The result must be received at a specific time within the window watchdog. The result from the host is then compared with the expected result that is stored in the CIC61508 NVM. Depending on the result of this comparison, the pass counter is incremented or decremented.

The sequencer has an SEQ SFR that defines the current request number (question). Upon a successful comparison of the current answer, the SEQ SFR is updated with the next request number. The request number and the corresponding 32-bit answer are configured in the NVM. The sequencer is provided with two parameters: minimum window period and maximum window period. The maximum window period is the window watchdog period, which is divided into the open window period and the closed window period. The minimum

window period is the closed window period. These two parameters are configurable in terms of heartbeats.

C. System Monitor

The system monitor is used for ensuring the accurate execution of application algorithms. For critical algorithms, the application code must have a redundant implementation. In practice, this may represent two instances of the same algorithm or preferably two differently ordered algorithms that perform identical calculations to yield identical numerical result. The application code must execute both algorithms and pass the results to the system monitor within a set time gap of each other. The system monitor then compares both results. The comparison performed may be in terms of equality, greater than, or less than, and a mask may be applied to the data to set the accuracy of the check.

The safety driver provides a set of compare functions, and comparison is executed on the PCP. This allows critical calculations to be performed by equivalent functions and the results to be compared. For example, the result of an ADC conversion may be passed to two different functions that use separate and, if possible, varying algorithms to perform a critical calculation. The results are then passed to the algorithm check function via an API call to the SFL driver. The SFL driver generates a message to the safety driver's system monitor on the PCP. The comparison is then performed. If the test fails, a noncritical error is generated, and the algorithm test counter is incremented.

D. Task Monitor

The safety driver task monitor is used for monitoring the running time of critical sequences of code (tasks) that are executed on the TriCore CPU. The task monitor ensures that the correct sequence of tasks is executed and that they run with the expected duration. During configuration, the developer must configure the task monitor with a sequence of tasks and their expected execution deadlines. The safety driver monitor checks that the messages arrive in order and that each critical section does not exceed its maximum allowed run time. If the critical sections are called out of order or a maximum run time is exceeded, a noncritical error is raised and the task execution counter is incremented.

V. Functional Safety Compliant EMB Control System

The overall control system has a cascaded architecture, as shown in Fig. 4. PI (proportional-integral) feedback control is applied to the force, speed, and current controllers. The position controller uses a proportional controller that determines the motor position command. The current, speed, and force controllers are designed to satisfy the required time domain specifications achieving the desired response performance [11].

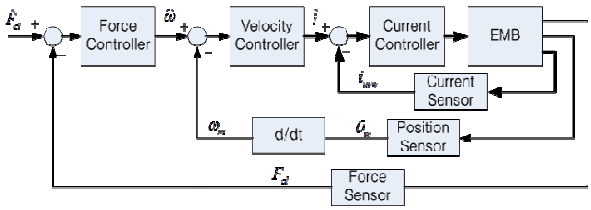

Fig. 4 Block diagram of EMB control system

The proposed architecture is shown in Fig. 5. The actuator ECU is implemented based on the Infineon TC1798, a 32-bit microcontroller with high-performance peripherals. The PWM signal for controlling the PMSM is generated using the CCU6. Analog signals from force, current, and rotary position sensors are converted to digital signals using the ADC module. Additionally, the commands for each controller can be transmitted from the CECU to the actuator ECU using the MultiCAN module. Additionally, TC1798 has two cores (TriCore 1.3.1 and PCP). TriCore is responsible for executing all safety-related applications covering all safety loops. PCP acts as the monitoring processor, covering execution integrity (mainly program sequence monitoring) of the main processor. The three components TriCore, PCP, and CIC61508 constitute a closed monitoring loop.

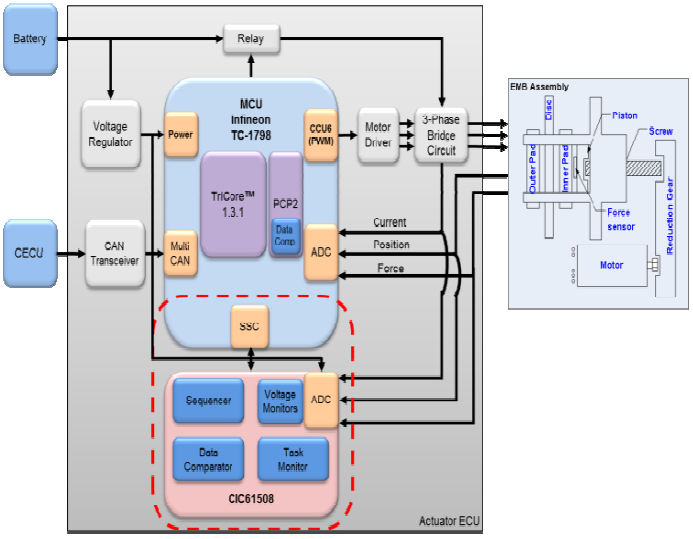

Fig. 5 Functional safety-compliant ECU architecture for EMB control system

The CIC61508 includes several modules such as four voltage monitors, a sequencer, data comparator, and task monitor. Through the voltage monitors, the CIC61508 is additionally capable of detecting under- and over-voltage from the supply to the monitored microcontroller, and the output voltages of each sensor for EMB actuator control, such as force, position, and current sensor.

The sequencer is responsible for monitoring the sequence of answers generated by the host. These answers are responses to the challenges initiated by CIC61508, and they verify the host

processor's integrity. The host responds to CIC61508 by sequentially sending a defined series of answers periodically within a defined timeframe. The sequencer monitor system verifies these answers against the static table stored in the CIC61508.

A data comparator compares two ADC result values from the force sensor for EMB force control delivered within a determined period to check for an equal, greater, or less than condition based on a predefined mask value and its scenario is as shown in Fig. 6. The task monitor uses a defined schedule table to check the dispatch of critical tasks running on the host microcontroller with predefined execution budgets. Such task deadline enforcements will allow, for example, force control, velocity control, and current control in EMB control systems and its scenario is as shown in Fig. 7. If CIC61598 detects error, the application error callback will be entered, CIC61508 reports error to application and cut off the power path of EMB system.

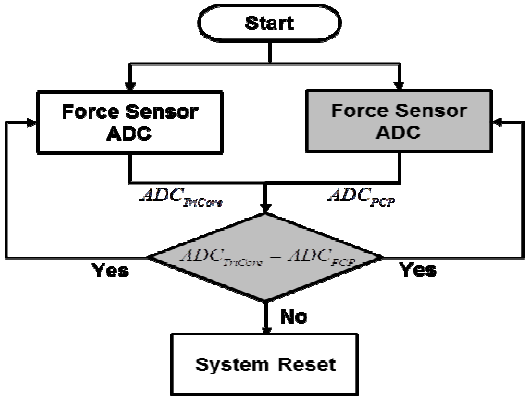

Fig. 6 The scenario of a data comparator

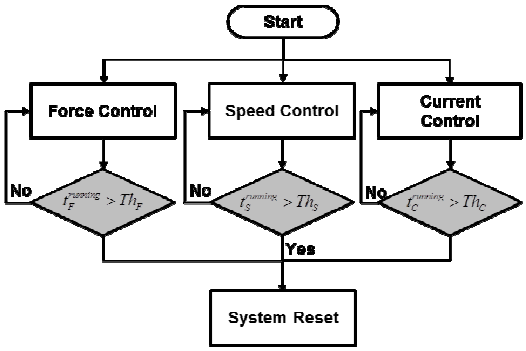

Fig. 7 The scenario of a task monitor

VI. Conclusion

In this paper, we proposed a functional safety-compliant ECU architecture and safety driver for EMB system. The ECU architecture was implemented based on TC1798 (TriCore and PCP) and CIC61508. The three components TriCore, PCP, and CIC61508 participate in a closed monitoring loop. Furthermore, the proposed safety driver consists of four monitor programs, namely, supply voltage monitor for detecting over- and under-voltage failures, sequencer for detecting clocking and program sequence failures of the MCU, system monitor, and task monitor for detecting latent faults in the safety mechanism. The proposed ECU architecture was applied to an EMB system and safety driver to be ASIL-D-compliant.

Acknowledgment

This research was supported by the MSIP (Ministry of Science, ICT&Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (NIPA-2013-H0301-13-2007) and the C-ITRC (Convergence Information Technology Research Center) support program (NIPA-2013-H0401-13-1008) supervised by the NIPA (National IT Industry Promotion Agency)

References

[1] Maron, C., Dieckmann, T., Hauck, S., and Prinzler, H.,"Electromechanical Brake System: Actuator ControlDevelopment System," SAE Technical Paper 970814, 1997,doi:0.4271/970814.
[2] Schwarz, R., Isermann, R., Böhm, J., Nell, J., andRieth P.,"Clamping Force Estimation for a Brake-by-WireActuator,"SAE Technical Paper 1999-01-0482, 1999, doi:10.4271/1999-01-0482.
[3] Zhai, Z. and Corbiere, T., "Achieving ASIL D for Microcontroller in Safety-Critical Drive-by-Wire System," SAE Technical Paper 2009-01-0759, 2009, doi:10.4271/2009-01-0759.
[4] ISO 26262-3 Road Vehicles - Functional safety – Part10: Guideline on ISO 26262, 2011.
[5] ISO 26262-3 Road Vehicles - Functional safety – Part3: Concept Phase, 2011.
[6] Cheon, J., Kim, J., Jeon, J., and Lee, S., "Brake By Wire Functional Safety Concept Design for ISO/DIS 26262,"SAE Technical Paper 2011-01-2357, 2011, doi:10.4271/2011-01-2357.
[7] Van Eikema Hommes, Q., "Review and Assessment of the ISO 26262 Draft Road Vehicle - Functional Safety,"SAE Technical Paper 2012-01-0025, 2012, doi:10.4271/2012-01-0025.
[8] Christiaens, S., Ogrzewalla, J., and Pischinger, S., "Functional Safety for Hybrid and Electric Vehicles,"SAE Technical Paper 2012-01-0032, 2012, doi:10.4271/2012-01-0032.
[9] ISO 26262-5Road Vehicles - Functional safety – Part5: Product development at the hardware level, 2011.
[10] Sundaram, P. and D'Ambrosio, J., "Controller Integrity in Automotive Failsafe System Architectures," SAE Technical Paper 2006-01-0840, 2006, doi:10.4271/2006-01-0840.
[11] Ki, Y., Ahn, H., and Cheon, J., "Fault-Tolerant Control of EMB Systems," SAE Int. J. Passeng. Cars - Electron. Electr. Syst. 5(2):579-589, 2012, doi:10.4271/2012-01-1795.