논문 2021-16-35

AUTOSAR 소프트웨어 기능안전 메커니즘 설계 사례연구: Shift-by-Wire 시스템

(Case Study on AUTOSAR Software Functional Safety Mechanism Design: Shift-by-Wire System)

금 대 현, 권 수 현, 이 재 성, 이 성 훈*
(Daehyun Kum, Soohyeon Kwon, Jaeseong Lee, Seonghun Lee)

Abstract : The automotive industry and academic research have been continuously conducting research on standardization such as AUTOSAR (AUTomotive Open System ARchitecture) and ISO26262 to solve problems such as safety and efficiency caused by the complexity of electric/electronic architecture of automotive. AUTOSAR is an automotive standard software platform that has a layered structure independent of MCU (Micro Controller Unit) hardware, and improves product reliability through software modularity and reusability. And, ISO26262, an international standard for automotive functional safety and suggests a method to minimize errors in automotive ECU (Electronic Control Unit)s by defining the development process and results for the entire life cycle of automotive electrical/electronic systems. These design methods are variously applied in representative automotive safety-critical systems. However, since the functional and safety requirements are different according to the characteristics of the safety-critical system, it is essential to research the AUTOSAR functional safety design method specialized for each application domain. In this paper, a software functional safety mechanism design method using AUTOSAR is proposed, and a new failure management framework is proposed to ensure the high reliability of the product. The AUTOSAR functional safety mechanism consists of memory partitioning protection, timing monitoring protection, and end-to-end protection. The fault management framework is composed of several safety SWCs to maintain the minimum function and performance even if a fault occurs during the operation of a safety-critical system. Finally, the proposed method is applied to the Shift-by-Wire system design to prove the validity of the proposed method.

Keywords : AUTOSAR, ECU, Functional Safety mechanism, Shift-by-wire, ISO26262

Ⅰ. 서 론

최근 자동차에는 안전, 편의, 인포테인먼트, ADAS (Advanced Driver Assistance Systems) 등 편리하고 안전한 기능들이 보편화 되고 있다. 이로 인해 자동차 네트워크에 연결된 전자제어장치 (ECU; Electronic Control Unit)와 내장된 소프트웨어의 크기와 복잡도 역시 함께 증가 되고 있다. 전기/전자 아키텍처와 ECU 소프트웨어의 복잡화는 전자제어장치 고장 발생 가능성을 높이고, 개발 과정에서 드는 설계/검증 노력과 비용도 증가하는 문제를 발생시킨다. 이런 문제를 해결하기 위해, 자동차 완성차, 부품회사, 반도체 회사 등의 관련 업계들은 ECU 시스템의 개발 효율성과 안전성을 강화를 목적으로, AUTOSAR (AUTomotive Open System ARchitecture) 규격 [1], ISO 26262 표준 [2] 등의 전자/전기 제어 시스템에 대한 표준 규격화 연구를 수행하고 있다. 특히 높은 수준의 안전성을 요구하는 안전필수 (Safety-Critical) 분야인 섀시, 엔진 도메인의 응용 시스템에서는 표준 규격화를 이용한 전자제어장치 개발 기술의 적용이 필수적이다.

AUTOSAR는 전 세계 자동차 업계들이 의해 제정된 자동차 표준 소프트웨어 플랫폼으로, MCU 하드웨어에 독립적인 계층화된 소프트웨어 구조를 갖는다. AUTOSAR를 이용해 ECU 개발 시 소프트웨어의 모듈화, 재사용성이 높아져 개발 비용을 절감할 수 있고, 제품의 신뢰성 향상이 가능하다. 현재 대부분 국내외 자동차 완성차 및 부품회사들은 AUTOSAR가 적용된 ECU를 양산하고 있다 [3]. 하지만 방대한 규격서 이해의 어려움과 사용자 컴포넌트 설정에 따른 시스템 성능 차이 등의 문제로 응용 시스템별로 최적화된 개발 방법에 대한 연구가 지속적으로 수행되고 있다 [4-7].

자동차 내 자동차 ECU의 급속한 증가로 인해 기능 안전성의 중요성이 강조되어지고 있다. 그리고 다수의 공급자에

*Corresponding Author (shunlee@dgist.ac.kr)
Received: Oct. 19, 2021, Revised: Nov. 17, 2021, Accepted: Dec. 4, 2021.
Daehyun Kum: DGIST (Principal Researcher)
Soohyeon Kwon: DGIST (Associate Researcher)
Jaeseong Lee: DGIST (Associate Researcher)
Seonghun Lee: DGIST (Principal Researcher)
※ 본 연구는 과학기술정보통신부에서 지원하는 DGIST 기관고유사업에 의해 수행되었습니다 (21-IT-02).

ⓒ IEMEK J. Embed. Sys. Appl. 2021 Dec. 16(6) 267-276
ISSN : 1975-5066
http://dx.doi.org/10.14372/IEMEK.2021.16.6.267

의한 ECU 개발이나 ECU의 복잡도 증가로 인해 개발 프로세스의 효율 제고 및 비용 절감을 위한 표준화에 대한 필요성 역시 커지고 있다. 이를 위해 ISO는 자동차 ECU의 오류로 인한 사고방지를 목적으로, 자동차 소프트웨어/하드웨어 개발자들이 동일한 안전 설계 기준을 따를 수 있도록 자동차 기능안전성 국제표준인 ISO26262 규격을 제정 했다 [2]. 이 규격에는 전자제어장치 개발 프로세스 모델과 함께 요구되는 활동, 유무형의 결과물, 그리고 개발과 생산에 사용되는 방식이 정의되어 있으며, 명세부터, 설계 구현, 통합, 검증, 인증, 생산까지 모든 자동차용 전기/전자 안전 관련 시스템의 제품 수명 전 주기에 걸쳐 적용 가능한 내용이 포함되어 있다. 자동차 안전 관련 요구사항을 지정하는 지표로 ASIL 등급을 사용하며, 소프트웨어 컴포넌트간의 간섭의 자유를 보장할 수 있는 다양한 설계 방법을 통해 ASIL 등급을 충족시키는 연구가 수행되고 있다 [2].

특히 자동차 대표적인 안전필수 시스템인 Drive-by-Wire 시스템 [8]에서는 결함을 최소화하기 위해, AUTOSAR 플랫폼 및 기능안전 설계에 대한 다양한 연구가 수행되고 있다. 안전필수 시스템에서 AUTOSAR를 설계하는 방법으로 듀얼 코어 MCU를 활용한 하드웨어 및 소프트웨어 설계에 대한 개념이 소개되었고 [9], AUTOSAR를 이용하여 안전관련 기능을 설계 시 필요한 고려사항이 제안되었다 [10]. AUTOSAR 기능안전 사양서 [11]를 따라, 타이밍 보호를 위한 설계와 응용 사례가 연구되었다 [12, 13]. 그리고 AUTOSAR 기반의 자동차 소프트웨어에서 발생할 수 있는 결함을 정의하고, 소프트웨어 개발과정에서 결함 주입 테스트 하는 방법이 제안되었다 [6, 14]. Steer-by-Wire 시스템에 AUTOSAR를 처음으로 적용하고 RP 환경에서 테스트가 수행되었고 [15], AUTOSAR 소프트웨어 할당 및 컴포넌트 구성을 최적화하기 위한 모델기반 방법론이 제안되었다 [16]. 또한 자동차 기능 안전 설계 방법론의 최근 연구를 정리하고 차세대 기능안전 표준인 ISO21448에 적응할 기능안전 설계 방법론의 미래 트렌드가 소개되었고 [17], Steer-by-wire 시스템의 기능안전 알고리즘 개발위한 프로세스 개발 및 검증하는 사례연구도 수행되었다 [18, 19].

지금까지 수행된 연구에서는 안전필수 시스템에서 AUTOSAR를 적용하는 방법 또는 기능안전 설계 프로세스 및 개발 방법을 제안되었거나, AUTOSAR에서 기능안전 설계 및 평가에 대한 연구를 각각 개별적으로 수행되었다. 하지만 AUTOSAR 기능 안전 설계의 경우는 시스템의 특성에 따라 기능 및 안전 요구사항이 달라지므로, 안전필수 시스템이 적용되는 응용분야별로 특화된 AUTOSAR 기능안전 설계 방법에 대한 연구가 필요한 실정이다.

본 연구에서는 자동차에서 대표적인 안전필수 시스템인 SBW (Shift-by-Wire) 시스템을 위한 AUTOSAR 기능안전 메커니즘 설계 방안을 제안한다. SBW 시스템은 최근 모든 자동차에서 거의 필수적으로 적용되는 전자식 변속시스템으로, 높은 신뢰성이 요구되는 제품이다. 본 논문에서는 SBW 시스템을 위한 AUTOSAR 표준 규격을 만족하는 기능안전 메커니즘을 설계하고, 높은 신뢰성 구현을 위해 고장관리 프레임워크를 새롭게 제안한다. 고장관리 프레임워크에는 소프트웨어 컴포넌트의 오류 검출, 기능 테스트 등의 기능을 수행하여, 응용 소프트웨어 컴포넌트의 검증이 가능하다.

본 논문은 II장에서 SBW 시스템과 AUTOSAR에 대한 개요를 소개하고, III장에서는 기능안전 메커니즘 설계 사례로 고장관리 프레임워크와 AUTOSAR 기능안전 메커니즘 설계방안에 대해 제안한다. IV장에서는 테스트 베드 환경에서 기능 수행 테스트를 통해 설계 방안을 검증하고, V장에서 결론을 맺는다.

II. SBW 시스템 및 AUTOSAR 개요

1. SBW 시스템 개요 [20]

SBW 시스템은 자동변속기의 P, R, N, D 모드 변환을 기계적인 변속 레버 대신 전자적인 구조로 대체한 시스템이다. 기존 시스템은 변속기와 변속 레버 사이의 기계적 결합 구조로 인해 변속 레버의 위치, 크기, 형상 등 설계의 자유도가 제한적인 단점을 가진다. 하지만, SBW 시스템은 표 1과 같이 레버식, 다이얼식, 버튼식 등의 형상을 가지며, 다양한 위치에 변속 레버를 장착이 가능하여 설계 자유도가 높아 공간 확보 측면에서 많은 장점을 가진다.

SBW 시스템은 배터리가 저전압 상태이거나, 저온/고온의 온도 등의 가혹한 조건에서도 정확한 변속시간, 위치 정확성, 도난방지 등의 복합 기능이 정상적으로 동작되어야 한다. 이를 위해서는 기구, 전기전자회로, 센서, 모터 등 전체 시스템이 상호 정교하게 연동되어야 되므로, 시스템 레벨에서 설계 및 검증이 필수적이며, 특히 복잡한 기능을 구현하는 소프트웨어 레벨에서 기능안전 설계가 필수적이다.

2. AUTOSAR 개요 [1]

2.1 AUTOSAR 플랫폼

AUTOSAR는 차량 ECU에 사용되는 임베디드 소프트웨어 플랫폼의 국제 표준 규격이다. ECU 소프트웨어를 개발 시에 표준화된 인터페이스, 운영 체제, 소프트웨어 모듈을 사용함으로써 소프트웨어의 재사용성 및 확장성 등을 향상시켜, 복잡한 소프트웨어를 빠르고 신뢰성 있게 개발 할 수 있다 [3]. AUTOSAR는 그림 1처럼 크게 클래식 플랫폼 (Classic Platform) 표준, 어댑티브 플랫폼 (Adaptive

표 1. SBW 시스템
Table 1. SBW System

Item	Lever type	Dial type	Button type
Image			
Feature	Conventional operation Easy position operation No. of parts ↑ Cost ↑	Space flexibility No. of parts ↓ Fixed eyes in operation New operation type	Space flexibility Simple operation Fixed eyes in operation New operation type

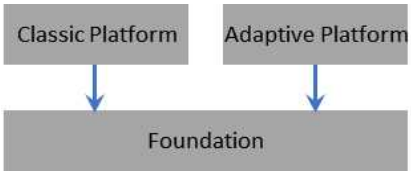

그림 1. AUTOSAR 표준 관계
Fig. 1. Dependencies of AUTOSAR standard

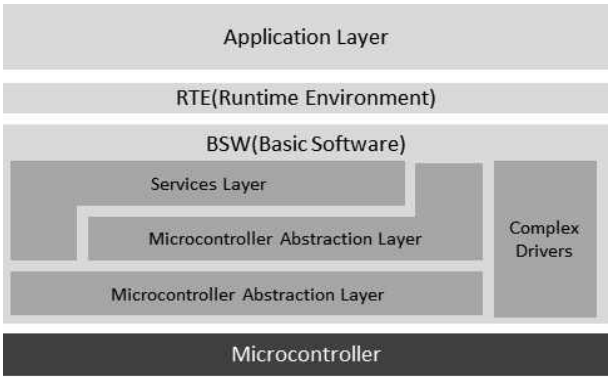

그림 2. AUTOSAR 클래식 플랫폼 아키텍처
Fig. 2. AUTOSAR Classic Platform layer architecture

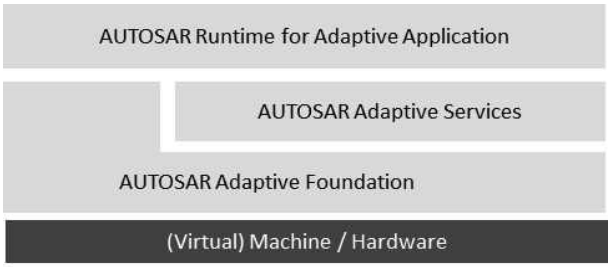

그림 3. AUTOSAR 어댑티브 플랫폼 아키텍처
Fig. 3. AUTOSAR Adaptive Platform layer architecture

Platform) 표준, 기반 (Foundation) 표준 총 3가지로 구성된다. 클래식 플랫폼 표준은 실시간성이나 안전을 중요시하는 임베디드 시스템을 위한 표준이고, 어댑티브 플랫폼은 자율주행 같은 고성능 연산이 필요한 임베디드 시스템을 위한 표준이며, 기반 표준은 두 AUTOSAR 플랫폼 표준간의 상호 호환성을 다루는 표준이다 [21].

AUTOSAR 클래식 플랫폼은 그림 2처럼 크게 응용 층 (Application Layer), RTE (Runtime Environment), 베이직 소프트웨어 (BSW (Basic Software)) 층, 총 3개의 층으로 구성된다. 베이직 소프트웨어 층은 다시 서비스 층 (Services Layer), ECU 추상화 층 (ECU Abstraction Layer), 마이크로컨트롤러 추상화 층(Microcontroller Abstraction Layer), 복합 드라이버 (Complex Drivers)로 구성된다. 이 층들은 하드웨어 추상화, 런어블 (Runnable)과 태스크의 스케줄링, 응용 소프트웨어 간의 통신, 진단, 그리고 안전 및 보안 등의 서비스를 수행한다 [3, 22].

AUTOSAR 어댑티브 플랫폼은 그림 3처럼 응용 런타임 (ARA (AUTOSAR Runtime for Adaptive Application) 층, 기반 (AUTOSAR Adaptive Foundation) 층, 서비스 (AUTOSAR Adaptive Services) 층, 총 3개 층으로 구성된다. 상위의 응용 런타임 층은 미들웨어 성격의 ara::com 을 통해 하위 기반 층과 서비스 층의 API를 호출해서 동작하며, 자율주행, 게이트웨이, 바디 도메인 제어기, 인포테인먼트 시스템 등의 응용 분야에 주로 사용 된다 [3, 22].

2.2 AUTOSAR 소프트웨어 기능안전 메커니즘 [11]

ISO26262 표준은 소프트웨어 컴포넌트 간의 간섭 결함의 원인을 메모리, 타이밍, 실행, 정보 교환 등으로 분류하고 있다. AUTOSAR는 소프트웨어 기능안전 메커니즘에서 결함의 방지, 검출, 완화를 위해 메모리 파티셔닝, 타이밍 모니터링, E2E (End-to-End) 보호 기능을 명세하고 있다.

메모리 파티셔닝은 낮은 ASIL 등급에 따라 개발된 소프트웨어 컴포넌트가 ASIL 등급이 높은 소프트웨어 컴포넌트의 메모리 영역에 잘못 액세스하여 발생하는 간섭을 방지하기 위해 사용되는 기술이다. ASIL 등급별로 별도의 메모리 영역에서 소프트웨어 컴포넌트를 실행하여 메모리 액세스 위반을 방지한다. 타이밍 모니터링은 소프트웨어 컴포넌트의 수행 시간을 관리하는 기술로, RTE와 기본 소프트웨어의 적절한 설계가 필요하다. E2E 보호는 통신 링크 내의 결함으로부터 데이터를 보호하는 기술로 송수신간의 데이터 교환 시 데이터 무결성을 보장하는 기술이다.

Ⅲ. AUTOSAR기반 소프트웨어 기능안전 메커니즘 설계 사례: SBW 시스템

본 장에서는 AUTOSAR 플랫폼을 기반으로 한 SBW 시스템의 안전과 관련 소프트웨어 컴포넌트와 베이직 소프트웨어 설계에 대해서 소개한다.

1. SBW 시스템 소프트웨어 아키텍처

SBW 시스템의 AUTOSAR 소프트웨어 아키텍처는 그림 4와 같이 응용 소프트웨어 (Application), RTE (Runtime Environment), 베이직 소프트웨어 (BSW)의 3가지 층으로 구성된다. 응용 소프트웨어는 5개의 소프트웨어 컴포넌트로

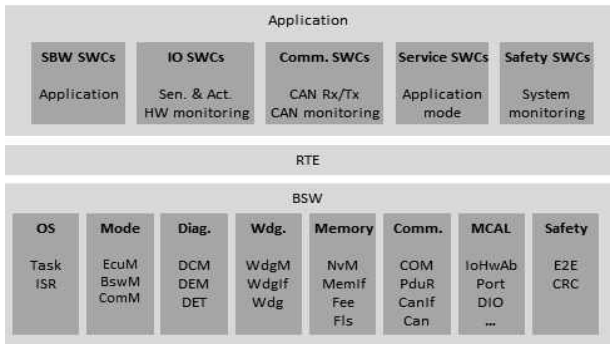

그림 4. SBW 시스템 AUTOSAR 소프트웨어 아키텍처
Fig. 4. Software Architecture of the SBW System

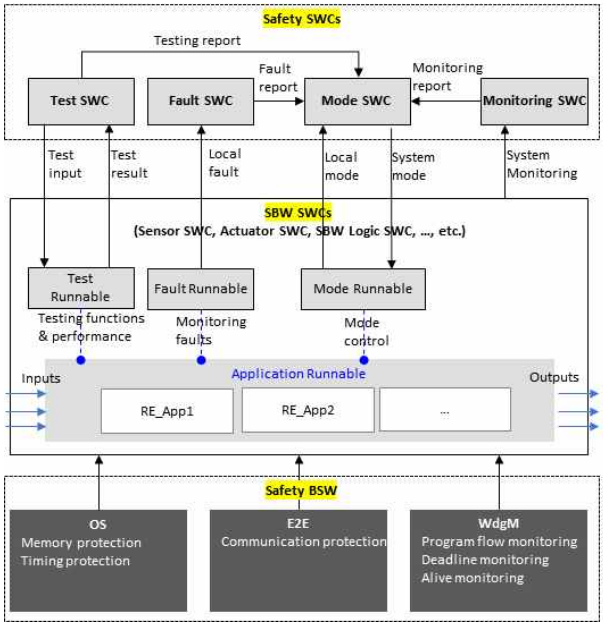

그림 5. 고장관리 소프트웨어 프레임워크
Fig. 5. Software Framework of the Fault Management

구성된다. SBW 소프트웨어 컴포넌트 (SBW SWCs)는 SBW 시스템 로직을 수행하고, 입출력 소프트웨어 컴포넌트 (IO SWCs)는 센서와 액추에이터의 하드웨어 입출력 제어를 수행한다. 통신 소프트웨어 컴포넌트 (Comm. SWCs)는 CAN 송수신, 차량의 상태에 따라 SBW 시스템의 모드를 제어하고, 서비스 소프트웨어 컴포넌트 (Service SWCs)는 메모리, 관리, 진단 등의 서비스를 수행하며, 마지막으로 안전관련 소프트웨어 컴포넌트 (Safety SWCs)는 시스템을 모니터링하고 시스템의 기능안전을 수행한다. RTE는 소프트웨어 컴포넌트 간의 데이터 교환 역할을 수행한다. 베이직 소프트웨어는 하드웨어 입출력, CAN 네트워크 메시지의 송수신, 메모리 읽기와 쓰기, ECU와 네트워크 모드 변환 그리고 기능안전을 위한 안전 모듈로 구성된다.

2. 고장관리 프레임워크 설계

SBW 시스템의 실행 중 발생할 수 있는 고장 검출 및 고장 대처를 위하여 고장관리 프레임워크를 제안하고, 시스템에서 고장이 발생하더라도 최소한의 기능 및 성능을 유지할 수 있도록 설계하였다. 그림 5는 제안한 SBW 시스템의 고장관리를 위한 소프트웨어 프레임워크이다. 안전 소프트웨어 아키텍처 (Safety SWCs)와 SBW 소프트웨어 컴포넌트 내부의 테스트 런어블 (Test Runnable), 고장 분석 런어블 (Fault Runnable), 모드 런어블 (Mode Runnable)이 고장관리를 수행하며, 베이직 소프트웨어의 OS, E2E, WdgM은 고장 검출을 위한 기능을 제공한다.

안전 소프트웨어 아키텍처 (Safety SWCs)는 고장을 검출하고 대처하기 위해, 테스트 소프트웨어 컴포넌트 (Test SWC), 고장 분석 소프트웨어 컴포넌트 (Fault SWC), 모니터링 소프트웨어 컴포넌트 (Monitoring SWC), 모드 소프트

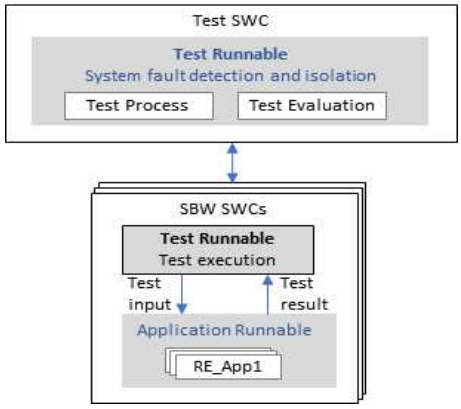

그림 6. 테스트 소프트웨어 컴포넌트
Fig. 6. Test Software Component

웨어 컴포넌트 (Mode SWC)로 구성된다. SBW 시스템의 소프트웨어 컴포넌트 (SBW SWCs)는 센서, 액추에이터, SBW 로직 등의 소프트웨어 컴포넌트들로 구성되며, 각 소프트웨어 컴포넌트는 기능별로 구분된 최소 실행 단위인 런어블로 구성된다. OS는 메모리와 타이밍 보호 기능을, E2E는 CAN 네트워크 메시지 및 소프트웨어 컴포넌트 사이의 통신 보호 기능을, 그리고 WdgM은 런어블의 실행순서, 마감시간, 실행시간 모니터링 기능을 각각 제공한다. OS, E2E, WdgM의 안전 메커니즘과 응용 소프트웨어 컴포넌트 내부의 고장검출 알고리즘을 바탕으로 소프트웨어 기능안전 설계를 하였다. 각 소프트웨어 컴포넌트는 그림 5와 같이 고장 관리를 위한 테스트 런어블 (Test Runnable), 고장 분석 런어블 (Fault Runnable), 모드 런어블 (Mode Runnable)과 기능 수행을 위한 응용 런어블 (Application Runnable)로 구성하였다.

그림 6의 테스트 소프트웨어 컴포넌트 (Test SWC)는 테스트 프로세스 런어블 (Test Process)과 테스트 결과 평가 런어블 (Test Evaluation)로 구성되고, 소프트웨어 컴포넌트들의 테스트 순서를 제어하고 테스트 결과를 기반으로 시스템의 상태를 평가하는 기능을 한다. SBW 소프트웨어 컴포넌트 내부의 테스트 런어블 (Test Runnable)은 기능 및 성능 테스트를 실행하고, 그 결과를 분석하여 고장 유무를 판단한다. 테스트 소프트웨어 컴포넌트는 시스템 시작 과정에서 일반적인 기능 및 성능 테스트와 차량 운행 중에 테스트하기 어렵거나 위험한 고장 주입 테스트를 수행한다. 센서, 액추에이터, 네트워크, 전원, 제어기 등의 이상 유무에 대해서 테스트 전용 모드에서 다양한 테스트를 진행한다.

고장 분석 소프트웨어 컴포넌트 (Fault SWC)는 소프트웨어의 실행 중에 발생하는 시스템의 고장의 위치 및 원인을 분석한다. 그림 7처럼 SBW 소프트웨어 컴포넌트 내부의 고장 분석 런어블 (Fault Runnable)은 소프트웨어 컴포넌트의 입력과 출력을 분석하여 고장을 판단하고 그 결과를 고장 분석 소프트웨어 컴포넌트로 보낸다. 고장 분석 소프트웨어 컴포넌트는 모든 응용 소프트웨어 컴포넌트에서 발생한 고장 데이터를 분석하여 시스템 전체의 고장을 진단한다.

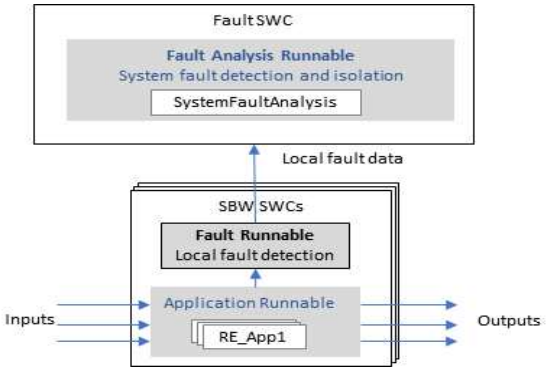

그림 7. 고장 분석 소프트웨어 컴포넌트
Fig. 7. Fault Software Component

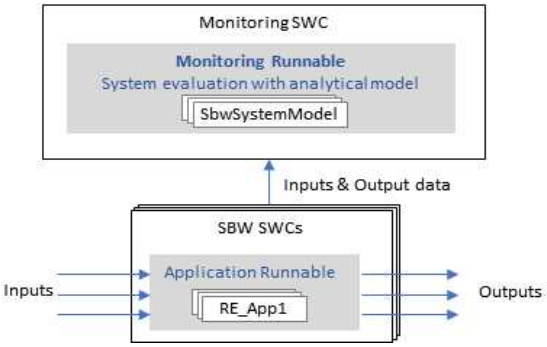

그림 8. 모니터링 소프트웨어 컴포넌트
Fig. 8. Monitoring Software Component

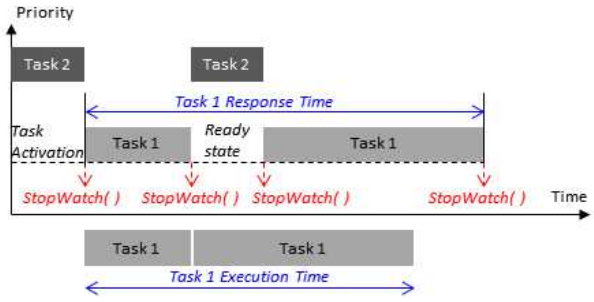

그림 9. 태스크 타이밍 보호
Fig. 9. Task Timing Protection

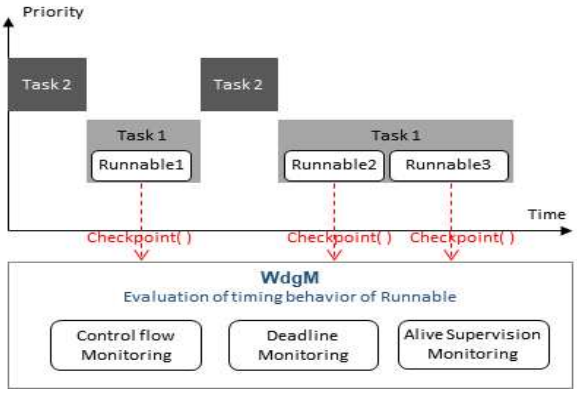

그림 10. 런어블 타이밍 모니터링
Fig. 10. Runnable Timing Monitoring

그림 8의 모니터링 소프트웨어 컴포넌트 (Monitoring SWC)는 시스템 기능 및 성능을 분석하고 센서, 액추에이터, 네트워크 등의 오류를 검출할 수 있는 다양한 시뮬레이션 모델로 구성된다. SBW 소프트웨어 컴포넌트들의 입력과 출력을 전달받아서 시스템 외부에서 독립적으로 실행되며 실시간으로 시스템 상태를 분석한다. 모니터링 소프트웨어 컴포넌트는 응용 소프트웨어 컴포넌트와 다른 별도의 메모리 영역에서 실행되며, IOC (Inter OsApplication Communicator)을 통해서 데이터를 주고받으며 시스템의 상태를 분석한다.

모드 소프트웨어 컴포넌트 (Mode SWC)는 시스템의 모드를 판단한다. 시스템 시작 전에는 테스트 소프트웨어 컴포넌트로부터, 시스템 동작 중에는 고장 분석 소프트웨어 컴포넌트와 시스템 외부의 모니터링 소프트웨어 컴포넌트로부터 시스템 상태 및 고장 정보를 모두 받아 시스템 모드를 판단한다. 만약 고장이 있다면 고장에 대처하기 위한 모드로 전환한다. 모드 런어블은 시스템의 상태에 따라서 내부 응용 런어블의 모드를 제어한다.

3. 타이밍 기능안전 설계

타이밍 기능안전은 OS와 WdgM을 이용하여 설계하였다. OS는 각 태스크들의 실행 시간과 실행 주기를 확인하고 WdgM은 런어블들의 실행 순서, 실행 주기, 실행 시간을 확인한다.

그림 9처럼 타이밍 보호를 위해서 OS에서는 태스크의 실행 시작과 실행이 중단되거나 종료될 때 스톱워치 타이머 (StopWatch)를 이용해서 시간을 측정한다. 스톱워치 타이머에 의해서 측정된 시간을 이용해서 실행 시간 (Execution Time)과 응답 시간 (Response Time)을 계산하고, 설정된 값을 초과한다면 오류가 발생하고 오류 대처 프로그램이 실행된다.

WdgM에서의 타이밍 보호 설계는 런어블 단위로 설계하며, 그림 10처럼 런어블 내부에 타이밍 보호가 필요한 부분에 체크 포인트 (Checkpoint)를 두어서 WdgM에 실행된 시간을 알려준다. 런어블 내부의 다양한 체크 포인트들의 실행 시간을 분석해서 런어블의 실행 순서가 정확하게 동작하는지를 확인하고 (Control flow Monitoring), 마감 시간이 초과하지 않는지를 확인하고 (Deadline Monitoring), 런어블의 실행 주기가 일정하게 유지되는 것을 확인한다 (Alive Supervision Monitoring). 만약 런어블의 실행 과정에서 오류가 발생하면 고장 복구 프로세스를 진행한다.

4. 메모리 기능안전 설계

메모리 기능안전 설계는 메모리 영역을 구분하여 서로 독립된 영역에서 소프트웨어를 실행시켜서 다른 메모리 영역에서 발생한 오류의 전파를 방지하도록 하였다. 소프트웨어 컴포넌트의 기능 및 안전도에 따라서 서로 다른 메모리 영

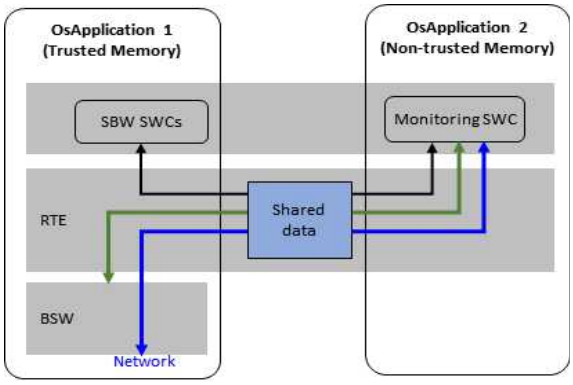

그림 11. 메모리 보호
Fig. 11. Memory Protection

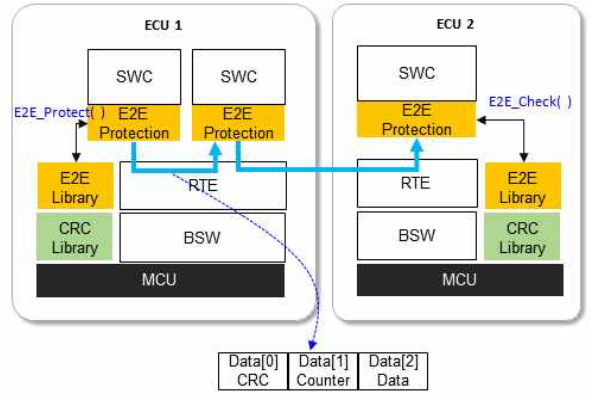

그림 12. 통신 보호
Fig. 12. Communication Protection

역에 할당하고 서로 간의 메모리 영역 접근을 제한하며, 메모리 영역 사이의 데이터 교환이 필요한 경우에는 공유 메모리 영역을 이용한다.

메모리 보호 기능은 AUTOSAR OS의 OsApplication과 MCU 하드웨어의 MPU (Memory Protection Unit) 이용하여 설계한다. 그림 11처럼 OsApplication1 메모리 영역에는 SBW 소프트웨어 컴포넌트 (Safety SWC)와 베이직 소프트웨어 (BSW)를 배치하고, OsApplication2 메모리 영역에는 SBW 시스템의 모니터링 소프트웨어 컴포넌트 (Non-Safety SWC)를 배치하여 독립된 메모리 영역에서 SBW 시스템 상태를 분석할 수 있도록 설계하였다. OsApplication2 영역의 모니터링 소프트웨어 컴포넌트는 공유 메모리 영역 (Shared data)을 이용하여 SBW 소프트웨어 컴포넌트의 데이터를 수신하고 시스템의 상태를 분석하여 그 결과를 송신한다.

5. 통신 기능안전 설계

ECU 내부 소프트웨어 컴포넌트 사이의 데이터 교환과 CAN 네트워크 메시지의 보호를 위해서 AUTOSAR에서 제공하는 E2E 통신 방식 중에서 E2E Profile 2를 사용하였다.

E2E 통신은 원본 데이터에 카운터를 추가하여 의도하지 않은 데이터의 손실, 삽입, 중복, 블록킹을 감지하고, CRC (Cyclic Redundancy Check)를 추가하여 데이터 왜곡 오류를 검출할 수 있도록 하였다. 그림 12처럼 카운터와 CRC를 원본 데이터에 추가하여 송신하고, 수신할 때 CRC와 카운터를 확인하여 데이터 전달 과정에서 발생할 수 있는 오류를 감지하고, 오류의 종류 및 반복 횟수에 따라서 복구 또는 대처하는 프로그램을 호출한다.

Ⅳ. 기능수행 테스트를 통한 설계 검증

1. SBW 테스트 환경

SBW 시스템 제어기의 MCU는 NXP사 S32K를 사용하였고 ETAS사의 AUTOSAR 솔루션을 활용하여 소프트웨어 컴포넌트 및 베이직 소프트웨어 개발하였다. 테스트 환경은 표 2 및 그림 13과 같이 SBW ECU의 내부 메모리 및 소프트웨어 실행 모니터링을 위해서 Trace32 디버거를 사용하였고, CAN 네트워크 송수신 모니터링을 위해서 벡터사 CANoe를 사용하였다.

표 2. 테스트 베드 구성
Table 2. Test Bed Configuration

Device	Description
SBW ECU	• SBW system controller (NXP S32K) • AUTOSAR platform (ETAS)
Trace32	• SBW ECU debugger
CANoe	• Network control and monitoring device (Vector)
Testing PC	• Test input/output control and analysis • Check source code and memory values
Power Supply	• DC Power supply

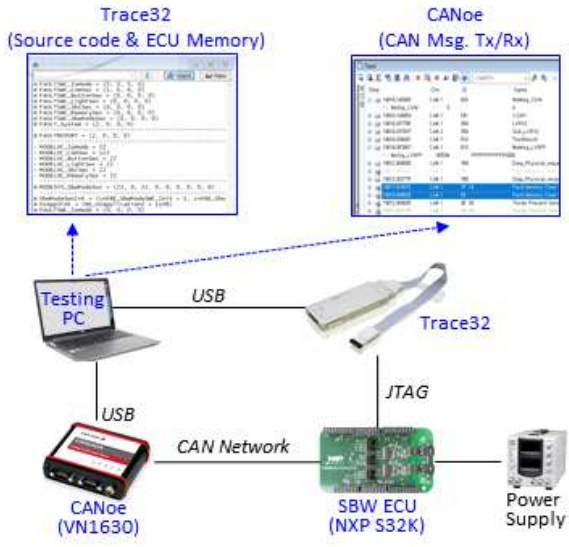

그림 13. 테스트 베드 구성
Fig. 13. Test Bed Configuration

테스트는 다음과 같이 진행되었다. 타이밍 기능안전 설계와 메모리 기능안전 설계 항목은 테스팅 PC에서 Trace32를 이용하여 테스트 입력을 제어하고 소스 코드 값 및 ECU 메모리 값을 확인하여 테스트 결과를 분석하였다. 통신 보호 테스트는 CANoe를 이용하여 오류 메시지를 ECU로 보내고, 수행 결과를 Trace32에서 확인하였으며, 오류가 발생하였을 경우에 고장 모드 변화 및 고장 대처 과정이 정상적으로 동작하는 것을 확인하였다.

2. 테스트 항목 및 테스트 결과

AUTOSAR 소프트웨어 기능안전 메커니즘 설계에 대한 테스트 항목은 표 3과 같다. AUTOSAR 기능안전 메커니즘 문서 [11]를 참고하여 총 4개 그룹의 세부 테스트 항목을 설계하였다.

고장관리 프레임워크의 테스트는 고장이 발생했을 경우 안전 모드로 전환되는지를 테스트 하였다. 그림 14는 CAN#1 네트워크에서 오류가 있을 경우, CAN#2 네트워크를 사용하도록 전환하는 테스트를 나타낸 것이다. 테스트 소프트웨어 컴포넌트에서 CAN#1 고장을 주입하여 고장을 감지하게 되면, 모든 소프트웨어 컴포넌트에서 CAN#1 대신 CAN#2 네트워크를 사용하도록 모드를 전환하게 된다. 또한 센서, 전원, 네트워크, 액추에이터 등의 고장 시에도 안전 모드로 전환되는 것을 테스트 하였다.

SBW 시스템은 OsApplication1에는 10개의 태스크, OsApplication2는 1개의 태스크로 구성되어 있으며, 각 태스크에 대해서 실행 시간 (Execution Time)과 실행 시간 간격 (Time Frame)을 설정하였다. 그림 15처럼 태스크의 타이밍

표 3. 테스트 항목
Table 3. Test Item

Item	Test Item of Function Safety Design
Fault Management Framework	• Check Fault Management SWC constitution • Check App. SWC runnable constitution • Check 입출력 Port constitution
Timing Protection	• Check OS General configuration • Check OsTask의 Timing Protection • Check WdgMGeneral configuration • Check WdgMSupervisedEntities configuration • Check WdgMConfigSet configuration • Check WdgMAliveSupervision configuration • Check WdgMDeadlineSupervision configuration • Check WdgMLocalStatusParams configuration • Check WdgM_MainFunction configuration • Check ProtectionHook implemetation • Check WdgM Port configuration • Check SWC Port configuration
Memory Protection	• Check EcuPartition generation • Check OsApplication generation • Check OsApplication2SWC generation • Check OS General configuration • Check ProtectionHook
Communication Protection	• Check E2E General configuration • Check E2E Interface configuration • Check SWC Port configuration

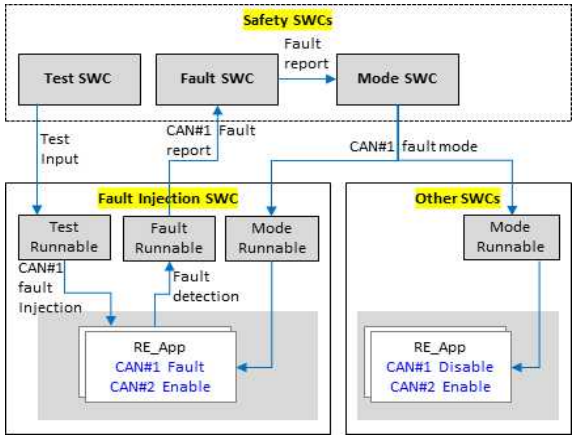

그림 14. 고장관리 프레임워크 테스트
Fig. 14. Framework of Fault Management Test

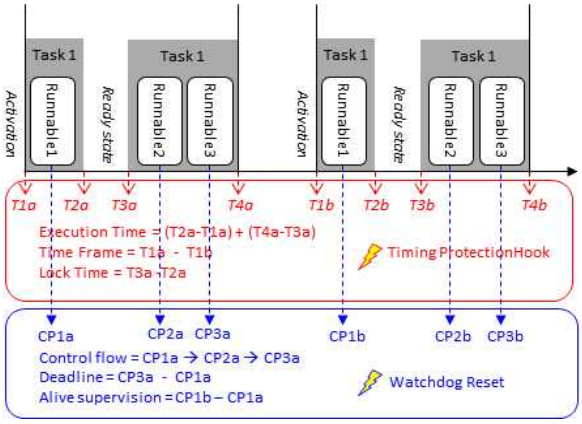

그림 15. 타이밍 보호 테스트
Fig. 15. Timing Protection Test

측정은 OS에서 Stopwatch 타이머를 별도로 설정하여 실행 시간 (Execution Time)과 실행 시간 간격 (Time Frame)을 측정하고 오류가 발생하면 TimingProtectionHook 함수를 실행하여 오류를 감지하고 대처할 수 있도록 하였다. 정상적인 실행 상태에서 오류의 발생 유무를 확인하고, 고의로 태스크의 실행 시간과 실행 시간 간격 오류를 주입하여 TimingProtectionHook 함수의 호출을 확인하였다. 런어블 타이밍 모니터링은 WdgM에서 이루어지며, 각 런어블의 실행 시 체크포인트 (CP)를 실행하여 런어블의 실행 순서, 마감 시간, 실행 간격을 측정하고, 고의로 오류를 주입하였을 때 시스템 리셋이 발생하는지 확인하였다. 태스크의 타이밍 측정은 80MHz 클럭을 사용하여 실시간 측정이 가능하며, 런어블의 타이밍 모니터링은 1ms 타이머를 이용하여 측정하고 10ms마다 오류를 판단한다. 테스트는 소프트웨어가 실행되는 동안 Trace32 디버거를 통해서 확인하였다.

SBW 시스템의 메모리 영역은 OsApplication1, OsApplication2 2개로 구성된다. OsApplication1 메모리 영역은 SBW 시스템의 제어에 관련된 태스크 (SBW Control Task)이며, OsApplication2 영역은 시스템 모니터링 태스크

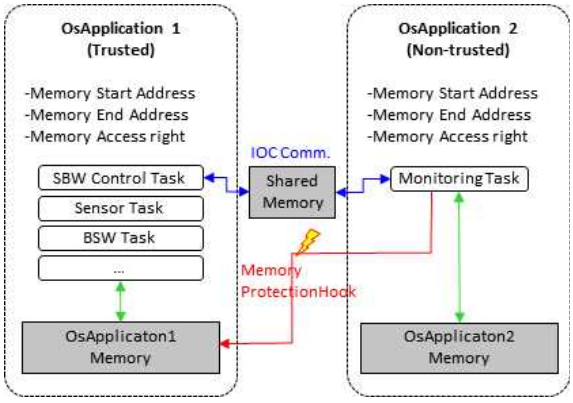

그림 16. 메모리 보호 테스트
Fig. 16. Memory Protection Test

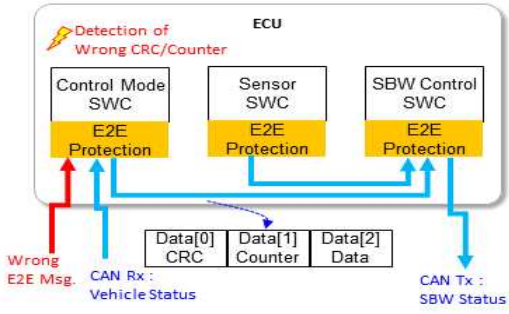

그림 17. 통신 보호 테스트
Fig. 17. Communication Protection Test

(Monitoring Task)이다. 그림 16처럼 MPU (Memory Protection Unit)를 이용하여 메모리 영역의 주소와 메모리 접근 권한에 관련된 설정하였다. OS에서 IOC (Inter OS Communicator) 설정을 하여 SBW 시스템의 입출력 데이터를 받는다. 메모리 보호 테스트는 IOC 통신을 이용하여 데이터를 정상적으로 주고받는 것과 OsApplication2에서 OsApplication1 영역을 직접 접근할 때 MemoryProtection Hook을 호출하여 오류를 감지하는지를 테스트 하였다.

SBW 시스템의 제어와 관련된 송수신 CAN 메시지와 소프트웨어 컴포넌트 사이의 데이터 교환 시에 E2E 통신을 이용하여 데이터를 보호한다. 차량 키 상태, 엔진 상태, 속도 등과 관련된 수신 메시지 5개와 SBW 시스템의 상태 출력 메시지 1개에 대해서 E2E 통신을 적용하였으며, 센서 입력, 제어 모드, 제어 명령 등 소프트웨어 컴포넌트의 주요 송수신 데이터에도 E2E 통신을 적용하였다. 그림 17처럼 데이터 송수신 과정에서 카운터와 CRC를 계산하여 정상적으로 주고받는 것과 고의로 데이터를 삽입하거나 데이터를 왜곡 시켰을 때 오류를 검출하는 지를 테스트 하였다.

제안한 설계 방안에 대한 테스트 결과를 정리하면 표 4와 같다. 첫째, 고장관리 프레임워크 테스트에서는 정상 상태에서 고장 테스트, 비정상 상태에서 고장 주입을 통한 고장 테스트, 감독 기능 테스트를 통해 양호 판정 결과를 확인했다. 둘째, 타이밍 기능안전 검증에서는 초기화, 실행 시간,

표 4. 테스트 결과
Table 4. Test result

Item	Test content	Result
Fault Management Framework	• Check Fault test in normal state • Check Fault Injection test in abnormal state • OsApplication2SWC - supervisor Check Functional test	• Pass • Pass • Pass
Timing Protection	• Check WdgM_Init( ) execution • Check inter-arrival time detection variable increasing • Check Execution Budget detection for 'OsTask_ASW' operation • Check Execution Budget detection for 'OsTask_OsApplication2' operation • Check Logical Supervision of E2E Com. variable • Check Logical Supervision of Mode. variable • Check Alive Supervision of E2E. variable • Check Alive Supervision of Mode. variable • Check Deadline Supervision of E2E Com.variable • Check Deadline Supervision of Mode variable	• Pass • Pass • Pass • Pass • Pass • Pass • Pass • Pass • Pass • Pass
Memory Protection	• Check Mpu_init( ) execution • Check SbwModeSWC - Write Enable/ Read Protection • Check SbwModeSWC - Read Enable/ Wire Protection	• Pass • Pass • Pass
Communica-tion Protection	• Check Initialization( ) execution • Check E2EP02_Wirte_IGNSW( ) → E2EP02_Read_IGNSW( ) • Check E2EP02_Wirte_SbwMode( ) → E2EP02_Read_SbwMode ( ) • Check E2EP02_Wirte_BtnStatus( ) → E2EP02_Read_BtnStatus( ) • Check E2E_DataTx_SbwMode, E2E_DataRx_SbwMode variable • Check E2E_DataTx_IGNSw, E2E_DataRx_IGNSw variable • Check E2E_DataTx_BtnStatus, E2E_DataRx_BtnStatus variable	• Pass • Pass • Pass • Pass • Pass • Pass • Pass

로직 감독 모드 및 변수 확인 등 기능 테스트를 통해 양호 판정 결과를 확인했다. 셋째, 메모리 기능안전에서는 초기화, 읽기/쓰기 보호 등 기능 테스트를 통해 양호 판정 결과를 확인했다. 마지막으로, 통신 기능안전 검증에서도 초기화, E2E 읽기/쓰기 및 변수 확인 등 기능 테스트를 통해 양호 판정 결과를 확인했다

V. 결 론

복잡해지는 차량용 전기/전자 아키텍처로 인해 ECU에 대한 신뢰성 확보 노력은 필수적이다. 특히 대표적인 차량 안전필수 시스템인 Drive-by-Wire 시스템에서는 전자제어 장치 결함을 최소화하기 위해, AUTOSAR 플랫폼 및 기능안전 설계에 대한 다양한 연구가 수행되고 있다. 이전 연구에서는 안전필수 시스템에서 AUTOSAR를 적용하는 방법 또는 기능안전 설계 프로세스 및 개발 방법을 제안하거나, AUTOSAR에서 기능안전 설계 및 평가에 대한 연구를 각각 개별적으로 수행되었다. 하지만 AUTOSAR 기능 안전

설계는 시스템의 특성에 따라 기능 및 안전 요구사항이 달라지므로, 안전필수 시스템이 적용되는 응용분야 별로 특화된 AUTOSAR 기능안전 설계 방법에 대한 연구가 필요한 실정이다. 따라서 본 논문에서는 SBW (Shift-by-Wire) 시스템을 위한 새로운 AUTOSAR 기능안전 설계 방안을 제안했다. AUTOSAR 표준 규격을 만족하는 기능안전 메커니즘을 설계하고, SBW 시스템의 높은 신뢰성 구현을 위해 고장관리 프레임워크를 새롭게 제안했다. 고장관리 프레임워크에는 소프트웨어 컴포넌트의 오류 검출, 기능 테스트 등의 기능이 포함되어, 시스템의 고장이 발생하더라도 최소한의 기능 및 성능을 유지할 수 있다.

제안된 SBW 시스템 고장관리 프레임워크는 테스트 소프트웨어 컴포넌트, 고장 분석 소프트웨어 컴포넌트, 모니터링 소프트웨어 컴포넌트, 모드 소프트웨어 컴포넌트로 구성된 안전 소프트웨어 아키텍처와 테스트 런어블, 고장 분석 런어블, 모드 런어블과 기능 수행을 위한 응용 런어블로 구성되어있다. 이 프레임워크를 이용해 시스템의 일반적인 기능/성능 테스트와 비정상적인 상황을 고려한 고장 주입 테스트가 가능하며, 고장의 위치 및 원인 분석이 가능하고, 시스템의 응용 소프트웨어 컴포넌트의 입출력을 통해 시스템의 상태 분석이 가능하다. 또한 이들 시스템 상태 및 고장 정보를 종합적으로 분석하여 시스템 모드를 판단하고, 고장 판별 시 고장에 대처하기 위한 모드 전환이 가능하다.

설계된 AUTOSAR 소프트웨어 기능안전 메커니즘의 타이밍 기능안전 설계를 통해 각 태스크와 런어블의 실행 시간, 실행 주기 등이 체크가 가능하여 실행 오류 판별 및 복구가 가능하다. 메모리 기능안전 설계를 통해 소프트웨어 안전도 및 기능에 따라 서로 다른 메모리 영역에 할당하고, 서로 간의 메모리 접근을 제한함으로써 소프트웨어 실행 시 발생되는 오류의 전파를 방지할 수 있었다. 통신 기능안전 설계를 통해 내부 소프트웨어 컴포넌트 사이의 데이터 교환과 CAN 네트워크 메시지 교환에서 발생할 수 있는 데이터 손실, 왜곡, 중복 등의 오류를 감지할 수 있었다.

제안된 설계 방법은 SBW시스템 테스트 베드에서 고장관리 프레임워크를 포함한 AUTOSAR 기능안전 메커니즘 설계 항목별로 테스트를 수행했으며, 각 항목별 API 동작 유무, 변수 값 확인 등을 통해, 설계한 항목들이 정상적으로 실행됨을 확인하였다. 이를 통해 SBW 시스템에서 AUTOSAR 일반적인 설계, 기능안전 메커니즘 설계, 그리고 고장 검출 및 판단 설계 방안의 타당성을 확인할 수 있으며, 향후 다른 자동차 안전필수 시스템을 위한 AUTOSAR 기능안전 메커니즘 설계 시 좋은 참고 문헌으로 활용될 수 있을 것이라 예상한다.

본 논문에서 제안한 AUTOSAR 소프트웨어 기능안전 설계 방안은 AUTOSAR 플랫폼에 대한 기본적인 설계 방법으로, 보다 향상되고 다양한 기능안전 설계를 위한 연구가 필요하다. 이를 위해 소프트웨어의 올바른 실행으로 제어흐름의 오류를 판단하는 논리적 감독 기술, 기능안전 측정 방안, 테스트 지표 도출, 응용 시스템 확장 등의 연구를 지속적으로 수행할 예정이다.

References

[1] http://www.autosar.org

[2] ISO 26262:2018. 2nd ed. Road vehicles - functional safety

[3] AUTOSAR, “AUTOSAR_EXP_Introduction,” Available online: https://www.autosar.org/fileadmin/ABOUT/ AUTOSAR_EXP_Introduction.pdf.

[4] D. H. Kum, S. H. Lee, G. M. Park, J. H. Cho, “Automated Testing System Using AUTOSAR XML,” Journal of IEMEK, Vol. 4. No. 4, pp. 156-163, 2009 (in Korean).

[5] G. M. Park, D. H. Kum, S. H. Lee, “Model-Based Development and Test Method for The AUTOSAR Embedded Software,” Journal of IEMEK, Vol. 4. No. 4, pp. 164-173, 2009 (in Korean).

[6] J. H. Park, B. J. Choi, “ASFIT: AUTOSAR-Based Software Fault Injection Test for Vehicles,” Electronics, Vol. 9. No. 5, pp. 850-22, 2020

[7] S. H. Lee, Y. J. Kim, D. H. Kum, S. H. Jin, “AUTOSAR Starter Kit for AUTOSAR Software Design,” Journal of IEMEK, Vol. 9. No. 2, pp. 87-98, 2014 (in Korean).

[8] R. Isermann, R. Schwartz, S. Stolzl, "Fault-tolerant Drive-by-wire Systems", IEEE Control Syst. Mag., Vol. 27, No. 5, pp. 64-81, Oct. 2002.

[9] S. P. Brewerton, F. Grosshauser, R. Schneider. “Practical use of Autosar in Safety Critical Automotive Systems,” SAE 2009 World Congress, Detroit, USA, April 2009.

[10] M. Graniou, H. Sivencrona, R. Svenningsson “Advantages and Challenges of Introducing AUTOSAR for Safety-related Systems,” SAE 2009 World Congress, Detroit, USA, April 2009.

[11] AUTOSAR, “AUTOSAR_EXP_FunctionalSafetyMeasures,” Available online: https://www.autosar.org/fileadmin/user_upload/ standards/classic/4-3/AUTOSAR_EXP_FunctionalSafetyMeasures.pdf.

[12] C. Ficek, N. Feiertag, K. Richter, "Applying the AUTOSAR Timing Protection to Build Safe and Efficient ISO 26262 Mixed-criticality Systems", ERTSS'2012, 2012.

[13] D. Eberhard, R. Schneider, F. Grosshauser, S. Brewerton, S. “Timing Protection in Multifunctional and Safety-Related Automotive Control Systems,” SAE 2009 World Congress, Detroit, USA, April 2009.

[14] T. Piper, S. Winter, N. Suri, T.E. Fuhrman, "On the Effective Use of Fault Injection for the Assessment of AUTOSAR Safety Mechanisms," Proc. of EDCC, pp. 85- 96, 2015.

[15] K. Chaaban, N. Rizoug, B. Barbedette, S. Saudrais, “Model-based Development of an Embedded

Steering-by-Wire system," Proceeding of 8th ISMA, UAE, 2012.
[16] A. Daghsen, K. Chaaban, S. Saudrais, "Software Function Allocation and Configuration of an Autosar Compliant System," SAE 2012 World Congress, Detroit, USA, April 2012.
[17] G. Xie, Y. Li, Y Han, Y. Xie, G. Zeng, "Recent Advances and Future Trends for Automotive Functional Safety Design Methodologies," IEEE Trans. Industrial Informatics, Vol. 16, No. 9, pp. 5629-5642, 2020.
[18] S. Lee, Y. Kwon, T. Hong, K. Park, "Development of Functional Safety Algorithm Based on Safety Clutch," KSAE Congress, Korea, 2017 (in Korean).
[19] Y. Noh, S. Bong, D. Kim, D. Lee, K. Park, "Based on ISO26262, Development and Validation of System Functional Safety Process for Steer-by-Wire System," KSAE Congress, Korea, 2016 (in Korean).
[20] K. Smirra, M. Ferst, T. Eiting, "Mechatronics for Shift by Wire - A Technical Challenge," SAE 2007 World Congress, Detroit, USA, April 2007.
[21] AUTOSAR, "AUTOSAR_TR_FoundationReleaseOverview," Available online: https://www.autosar.org/fileadmin/user_upload/ standards/foundation/19-11/AUTOSAR_TR_FoundationReleaseOverview.pdf
[22] AUTOSAR, "AUTOSAR_EXP_SafetyOverview," Available online: https://www.autosar.org/fileadmin/user_upload/ standards/adaptive/ 19-11/AUTOSAR_EXP_SafetyOverview.pdf

Daehyun Kum (금 대 현)

2001 Automotive Engineering from Keimyung University, Daegu, Republic of Korea (B.S.)
2003 Automotive Engineering from Keimyung University, Daegu, Republic of Korea (M.S.)
2011 Electronic Engineering from Kyungpook National University, Daegu, Republic of Korea (Ph.D. Course Completion)
2005~ Division of Automotive Technology, Convergence Research Institute, in DGIST (Principal Researcher)
Career:
2003~2005 LG Electronics, Researcher
2016~2017 University of Michigan, Visiting Researcher
Field of Interests: Automotive Embedded System, Software Safety, Cybersecurity, Automotive Control System
Email: kumdh@dgist.ac.kr

Soohyeon Kwon (권 수 현)

2006 Electronic Engineering from Kyungpook National University, Daegu, Republic of Korea (B.S.)
2008 Electronic Engineering from Kyungpook National University, Daegu, Republic of Korea (M.S.)
2010 Electronic Engineering from Kyungpook National University, Daegu, Republic of Korea (Ph.D Course Completion)
2011~ Division of Automotive Technology, Convergence Research Institute, in DGIST (Associate Researcher)
Career:
2010-2011 Electrical and Computer Engineering, in Colorado State University (Visiting Researcher)
Field of Interests: Automotive embedded systems, AUTOSAR
Email: shkwon@dgist.ac.kr

Jaeseong Lee (이 재 성)

2013 Embedded System Engineering from Daegu University, Gyeongsan, Republic of Korea (B.S.)
2015 Information and Communication Engineering from Daegu University, Gyeongsan, Republic of Korea (M.S.)
2005~ Division of Automotive Technology, Convergence Research Institute, in DGIST (Associate Researcher)
Field of Interests: Automotive embedded systems, AUTOSAR
Email: jaeseonglee@dgist.ac.kr

Seonghun Lee (이 성 훈)

1996 Electronic Engineering from Kyungpook National University, Daegu, Republic of Korea (B.S.)
1998 Electronic Engineering from Kyungpook National University, Daegu, Republic of Korea (M.S.)
2007 Electronic Engineering from Kyungpook National University, Daegu, Republic of Korea (Ph.D.)
2005~ Division of Automotive Technology, Convergence Research Institute, in DGIST (Principal Researcher)
Career:
2002~2005 Division of Main Battle Tank, in Agency for Defense Development (Senior Researcher)
2013~2014 Mechanical Engineering, in University of Michigan- Dearborn (Visiting Researcher)
2020~2021 Infineon Technologies Korea (Technical Advisor)
Field of Interests: Automotive embedded systems, Cyber-security, MCU application design, AUTOSAR
Email: shunlee@dgist.ac.kr