# 가상 통합 통신 시뮬레이션을 위한 LIN FMU 구현 제안

이하림[1], 곽준호[2], 김형래[3], 조정훈[4]
[1]경북대학교 전자전기공학과 석사과정
[2]경북대학교 전자전기공학과 박사과정
[3]경북대학교 전자전기공학과 박사과정
[4]경북대학교 전자전기공학과 교수
hw05165@knu.ac.kr, junho7513@knu.ac.kr, hrsin95@knu.ac.kr, jcho@knu.ac.kr

# Proposal for the implementation of LIN FMU in virtual integrated communication simulation

Ha-Rim Lee[1], Jun-Ho Kwak[2], Hyeong-Rae Kim[3], Jeong-Hun Cho[4]
[1234]Dept. of Electronic and Electrical Engineering, Kyungpook National University

요 약

차량용 소프트웨어 개발에서는 시뮬레이션 절차가 중요한 단계를 차지한다. 특히, 차량에서의 소프트웨어와 하드웨어 사이의 밀접한 종속성은 종종 개발 과정을 지연시키는 주요 요인으로 작용한다. 이에 개발자들은 가상 ECU 기술을 활용하여 소프트웨어를 하드웨어로부터 독립시키는 방안을 채택하고 있다. 이러한 가상 ECU 시스템은 다양한 개발 도구에서 생성된 모델을 FMU 형태로 변환, FMI 표준을 통해 데이터를 손쉽게 교환할 수 있도록 한다. 본 연구는 이와 같은 통합 시뮬레이션 환경 내에서 LIN 통신을 지원할 수 있는 LIN FMU의 구현 방법을 다룬다.

## 1. 서론

최근 차량용 소프트웨어 개발에서는 ADAS, 자율주행, 다양한 편의 기능 등의 많은 기능들을 지원하도록 바뀌고 있다. 이러한 기능들은 차량 내에서도 많은 데이터를 요구하기에 높은 대역폭과 빠른 속도를 가진 통신 기술을 필요로 하고 이에 맞추어 CAN FD, Ethernet과 같은 통신 네트워크를 지원하도록 한다. 하지만 여전히 비용적인 측면으로 인해 CAN과 LIN과 같이 비용이 상대적으로 낮은 네트워크 역시 차량에서 빠질 수 없다. CAN 통신 네트워크는 현재 차량 내에서 가장 많이 사용되는 통신으로 2개의 와이어에 차동 전압을 이용하여 외부 노이즈에 강하고 ECU 노드 간의 연결이 Point To Point 방식이 아닌 여러 개의 노드가 하나의 채널을 공유하는 버스를 사용하여 기존의 방식보다 와이어 사용량이 줄어들어 비용도 절감된다. 하지만 그럼에도 불구하고 CAN 통신 네트워크 만큼의 안정성 또는 속도를 필요로 하지 않는 곳에선 더 비용을 절감하기 위해 LIN 통신 네트워크를 사용한다. 예를 들어 실내 온도 조절 장치, 시트, 도어, 사이드 미러와 같은 안정성과 속도 측면에서 덜 중요한 편의 기능들에 LIN 통신 네트워크를 적용한다. LIN 통신 네트워크는 하나의 와이어를 사용하며 CAN Transceiver 라는 별도의 송수신기를 사용하지 않고 UART 통신에 쓰이는 직렬 통신 인터페이스(SCI)를 사용하여 추가적인 비용을 절감한다. 그러므로 CAN 통신과 더불어 LIN 통신은 차량 소프트웨어 개발에서 여전히 중요한 역할을 한다. 따라서 앞으로 가상 ECU 기술을 위해선 LIN 통신 또한 가상 시뮬레이션 환경에서도 검증되어야 한다.

본 연구에서는 Functional Mook-up Interface, FMI를 적용하기 위해 Simulink에서 개발한 하나의 ECU 모듈을 FMU로 내보내어 다른 FMU뿐만 아니라 다른 여러 외부 모듈과 통신할 수 있도록 구현하였다.

본 연구의 2장인 관련 연구에서는 구현에서 활용한 Simulink와 LIN 통신 프로토콜의 정의를 정리하고 3장에서는 LIN FMU에 대한 구현 방법을 제안한다. 마지막으로 4장에서는 제안된 LIN FMU의 활용방법과 기대 효과를 결론으로 마무리 짓는다.

## 2. 관련 연구

### 2.1 Simulink를 이용한 모델 기반 설계 (MBD)

Simulink[1]는 MATLAB 환경 내에서 운용되는 시

뮬레이션 및 모델 기반 설계 플랫폼으로, 다양한 엔지니어링 응용 분야에서 복잡한 시스템의 설계와 시뮬레이션을 지원한다. Simulink를 활용한 모델 기반 설계는 시스템의 개발 과정을 효율화하고, 설계 오류를 초기 단계에서 발견하여 수정할 수 있는 효과적인 방법을 제공한다.

Simulink는 GUI 인터페이스를 통해 시스템의 구성 요소를 시각적으로 모델링할 수 있게 해준다. 이는 개발자가 복잡한 수학적 모델을 코드로 직접 작성하지 않고도, 다양한 시스템의 동작을 설계하고 검증할 수 있도록 돕는다.

또한 모델로부터 직접 실행 가능한 코드뿐만 아니라, FMI (Functional Mock-up Interface) [2] 표준을 지원하는 코드를 자동으로 생성할 수 있는 기능을 제공한다. 이는 소프트웨어 개발 시간을 단축하고, 오류 가능성을 줄이며, 다른 시뮬레이션 플랫폼과의 호환을 가능하게 한다. 특히, FMI export 기능은 다른 시스템 시뮬레이션 소프트웨어와의 통합을 용이하게 하여, 시스템의 상호 운용성을 향상시키고, 통합 시험 및 검증 과정에서의 유연성을 제공한다.

### 2.2 LIN 통신 프로토콜

LIN (Local Interconnect Network) [3] 통신 프로토콜은 차량 내 통신 네트워크에서 각 ECU(전자 제어 장치) 간의 데이터 교환을 단순화하기 위해 설계된 통신 프로토콜이다. 이 프로토콜은 단일 와이어 버스를 통해 최대 16개의 노드를 연결할 수 있으며, Master-Slave 방식을 사용하여 통신한다. 이는 Multi Master 방식을 사용하는 CAN 통신 프로토콜과 대조된다. LIN 통신은 별도의 송수신기 모듈 없이 기존의 직렬 통신 인터페이스를 사용하기 때문에, Master 노드가 먼저 프레임 헤더를 송신하고, 해당 아이디를 갖는 메시지를 보내야 하는 Slave 노드가 프레임 리스폰스를 통해 응답하는 방식으로 충돌 없이 메시지를 전달한다.

LIN 통신 프로토콜은 사전에 정의된 타임 테이블에 따라 통신을 진행하기 때문에, 재전송을 허용하지 않는 결정적인 통신 방식을 채택하고 있다. 이는 차량 내에서 실시간성과 신뢰성을 보장해준다.

차량 내 CAN 네트워크는 특정 시간 간격마다 정적으로 메시지를 처리하는 반면, LIN은 미리 정의된 타임 테이블에 따라 통신한다. LIN은 타임 테이블의 주기가 종료된 후 즉시 다시 시작되므로 항상 버스가 점유된 상태가 되고, CAN은 특정 시간 간격 안에 모든 메시지가 처리되고도 남은 시간은 버스가 점유되지 않게 된다. 이러한 차이는 특히 ECU 노드의 수가 적고 메시지 수가 제한적인 경우 LIN의 접근 방식이 더 효율적일 수 있음을 의미한다. 예를 들어, CAN의 최대 전송 속도는 1Mbit/s에 달하지만, 실제 버스 점유 시간을 고려하면 LIN의 최대 20Kbit/s 속도가 더 적합할 수 있다. LIN은 네트워크 정의된 메시지가 15개 이하일 때 특히 효율적이며, 이는 외부 노이즈에 덜 민감하고, 버스 점유율이 높은 상황에서 유리하다.

LIN 통신에서는 마스터 노드가 보내는 프레임 헤더는 Sync-break field를 시작으로, 연속된 5개의 라이징 엣지를 포함하는 Sync field, 그리고 아이디 정보로 구성된다. 슬레이브 노드는 최대 8바이트의 데이터와 함께 체크섬을 포함한 프레임 리스폰스를 송신하여 데이터의 무결성을 보장한다. LIN 통신 프로토콜은 다양한 유형의 프레임을 사용하여 통신할 수 있다. 총 4가지의 프레임 유형이 있지만 주로 사용되는 것은 Unconditional Frame과 Diagnostic Frame으로, 차량에서는 정적으로 정의된 통신을 요구하기에 이들을 사용한다.

## 3. LIN 통신 네트워크를 위한 FMU 구현 방법

차량용 LIN 통신 시뮬레이션을 효과적으로 수행하기 위해 Simulink를 활용하여 ECU 노드들과 LIN 버스를 모델링하였다. 이 모델들은 LIN 통신 프로토콜을 반영하도록 설계되었으며, Simulink에서 지원하는 FMU export 기능을 이용해 시뮬레이션에 필요한 동작을 재현할 수 있다. 모델 기반 설계 접근법을 적용함으로써, 각 ECU 노드에서의 LIN 통신 기능을 체계적으로 분석하고, 이를 기반으로 FMU를 생성하여 시뮬레이션의 정확성을 극대화하였다.

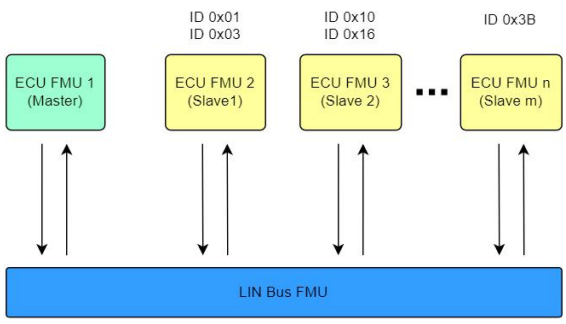
LIN 통신 프로토콜을 따라, 마스터는 Frame Header를 송신, 슬레이브는 Frame Response를 송신 또는 수신한다. 이러한 동작을 구현하기 위해, LIN Bus FMU는 다수의 입력 포트와 하나의 출력 포트를 포함하도록 설계되었다. FMU에서 하나의 입력 포트에서는 다른 FMU의 출력과 일대일 대응만 가능하기 때문에, 여러 입력 포트를 통해 여러개의 노드의 출력을 입력 받을 수 있다. 이 구조와 함께 입력되는 메시지의 길이를 계산하여 각 노드가 버스를 점유하는 시간을 정확하게 계산하고, 이 시간이 종료되면 받은 메세지를 모든 노드에게 Broadcast한다.

실제 LIN 버스와 시뮬레이션 환경 간의 차이를 고려하여 즉, ECU와 일반 PC 간의 클럭 속도 차이를 고려하여, LIN Bus FMU는 ECU 노드가 버스를 점유하는 시간을 대신 계산하고 메시지를 효과적으로 전송하도록 최적화하였다. 각 ECU 노드는 자신의 메시지를 LIN Bus FMU에 전달하며, LIN Bus FMU는 이 메시지들을 계산된 시간 이후에 모든 노드로 재전송한다.

LIN Bus FMU에 연결될 각 노드에 해당하는 ECU FMU들은 프레임을 구분하기 위해 프레임 헤더의 Sync-break 필드의 특성을 통해 프레임을 식별하는 방식을 택하였다. LIN 통신에서 각 바이트는 시작 비트 '0'와 종료 비트 '1'을 함께 총 10개의 비트를 순차적으로 전송하게 되지만, 프레임 헤더의 시작을 알리기 위한 Sync-break 필드는 최소 13개의 dominant bit '0'으로 구성되어 다른 ECU들이 이를 인식할 수 있게 한다. 각 ECU FMU들은 이 정보를 활용하여 프레임 헤더를 구분하고, 다음에 이어질 각 메시지에 맞는 프레임 리스폰스를 데이터 필드와 체크섬 필드로 나누어 송수신한다.

또한 가상 환경과 외부 환경 간의 원활한 통신을 지원하기 위해 Wrapper FMU를 개발하였다. 이 FMU는 POSIX 기반 소켓 통신을 사용하여 외부에서 수신된 메시지를 분석하고 처리할 수 있다. 이 구조를 통해 실제 환경에서의 데이터와 가상 환경 내부의 FMU 간의 데이터 교환이 효과적으로 이루어질 수 있도록 설계되었다.

현재 FMU들을 함께 시뮬레이션하는 Co-simulation 환경은 오픈소스인 MasterSim으로 FMI는 가상 시뮬레이션 환경에서 스텝(step) 단위로 동작한다. 이는 실시간으로 동작하는 실제 실물 보드와의 성능 비교를 하기에는 제한이 있다. 이를 위해서는 추후에 실제 보드를 에뮬레이션하고 에뮬레이션된 보드를 FMU와 동일하게 스텝 단위로 동작할 수 있도록 기술 구현이 필요하다.




(그림 1) LIN 가상 시뮬레이션 구성.

## 3. 결론

LIN 통신 기술과 FMU 기반의 시뮬레이션 통합은 차량용 소프트웨어 개발에서 중요한 진전을 나타낸다. 본 연구에서 제안한 LIN FMU는 차량 내 편의 기능을 위한 통신 네크워크인 LIN 통신 프로토콜을 가상 환경에서 시뮬레이션 할 수 있게함으로써 효율적인 개발과 시스템의 검증을 동시에 제공한다. 또한, 이 기술은 차량 내 다른 시스템과의 원활한 통합을 가능하게 하며, 향후 차량 통신 기술 발전에 중요한 기여를 할 것으로 기대된다. 따라서, 이 연구는 차량용 소프트웨어 시스템의 설계 및 구현 방법론에 새로운 방향을 제시하며, 향후 연구와 개발에 대한 기초를 마련할 것이다.

## ACKNOWLEDGMENT

이 논문은 2023 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.1711160343, 차량 ECU 응용소프트웨어 개발 및 검증자동화를 위한 가상 ECU 기반 차량레벨 통합 시뮬레이션 기술개발).

## 참고문헌

[1] Conrad, M., & Mosterman, P. Model-Based Design Using Simulink - Modeling, Code Generation, Verification, and Validation. Hoboken, NJ: Wiley. 2013.

[2] Blochwitz, Torsten, "Functional mock-up interface for model exchange and co-simulation," https://fmi-standard.org/ downloads/, 2014.

[3] Krishnamoorthy, Ramesh, Bharatiraja Chokkalingam, and Josiah Lange Munda. 2023. "Design of Fault-Tolerant Automotive Gateway Architecture Using MC9S12XDP512 Microcontroller Device" Energies 16, no. 16: 5923. 2023.