https://doi.org/10.7840/kics.2016.41.12.1959

매핑 테이블 기반의 자동차용 게이트웨이 설계

오 세 춘◆°, 김 의 룡*, 김 영 곤*

Design of the Automotive Gateway Based on a Mapping Table

Se-Chun Oh◆°, Eui-Ryong Kim*, Young-Gon Kim*

요 약

최근의 자동차 내부에는 수많은 ECU(Electronic Control Unit)들이 사용되고 있으며 또한 각각의 ECU들은 그 특성에 맞게 다양한 종류의 네트워크들에 연결되어 사용되고 있다. 따라서 다양한 네트워크 간의 효율적인 데이터 교환이 매우 중요한 요소로 등장했는데 이러한 서로 다른 네트워크간의 데이터를 교환해 주는 기능을 담당하는 ECU가 게이트웨이이다. 본 논문은 이러한 게이트웨이 설계에 있어서 서로 다른 네트워크 간의 데이터 교환의 효율성을 높이기 위한 매핑 테이블의 구조 및 출력되는 데이터들의 우선순위를 임의로 조절하는 기능을 갖는 새로운 게이트웨이 알고리즘의 구조를 제안하는데 그 목적이 있다. 또한 제안된 게이트웨이 구조는 특정 네트워크에서의 데이터 입력이 복수개의 서로 다른 네트워크로 동시에 변환 및 전달이 가능하고 전체 데이터 구조가 변경되더라도 게이트웨이 내부의 테이블 만을 변경하면 쉽게 적용되는 장점을 가지고 있다.

Key Words : Gateway, FlexRay, CAN, IVN, Automotive

ABSTRACT

The recent automobiles, a number of ECU inside the vehicle has been used. Also, each ECU is connected to different types of networks in accordance with the characteristics. Therefore, efficient data exchange between discrete network has emerged as a very important element. The gateway is responsible for the ability to exchange data between discrete network. In this study, we propose the new gateway algorithm to provide the structure of the mapping table to improve the efficiency of data exchange between discrete network. Also it provides a structure of a new gateway algorithm with a function of adjusting the priority of the data to be transmitted to another network arbitrarily. Moreover, the proposed gateway structure may simultaneously convert the transmission data input from a particular network to multiple networks. Another advantage is easy to change the entire data structure only if we change the table structure in the gateway.

Ⅰ. 서 론

전자산업의 발달로 근래의 자동차는 수많은 ECU(Electronic Control Unit, 전자제어장치)를 사용하고 있다. 초창기에 적용된 ECU들은 단순히 자동차 내의 수많은 센서, 스위치, 액추에이터(모터, 램프 등) 간을 연결하는 전기배선(wiring harness)들을 줄이기 위한 목적으로 사용되었으나 근래에는 엔진 제어와 변속기 제어 등의 성능 향상, 섀시(chassis) 제어, ABS, TCS 등을 통한 차량 안정성 확보는 물론 기타 바디(body) 제어 및 편의장치로의 적용이 활발히 이루어졌다. 이렇게 다양한 기능들을 처리하기 위해서

◆° First and Corresponding Author : Korea Polytechnic University Department of Computer Engineering, sc.oh@kpu.ac.kr, 정회원
* Korea Polytechnic University Department of Computer Engineering, euiryong@kpu.ac.kr, 정회원
* Korea Polytechnic University Department of Computer Engineering, ykkim@kpu.ac.kr, 종신회원
논문번호 : KICS2016-09-280, Received September 30, 2016; Revised October 21, 2016; Accepted October 21, 2016

많은 ECU들이 차량에 적용되면서 이들 ECU들은 그 특성에 따라 다양한 종류의 차량용 네트워크(IVN : In-Vehicle Network)로 연결되어져 있다[1][2]. 차량의 바디 제어 및 파워트레인(powertrain) 제어용 ECU들 간의 통신을 위해서 CAN이 등장하게 되었으며 근래에는 많은 자동차들이 새시 제어 부문까지도 기본적인 표준 네트워크로 CAN을 사용하고 있다. CAN의 도입으로 상승되는 제조원가를 절감하기 위해 일부 단순한 기능을 하는 ECU들은 낮은 가격의 LIN을 사용하여 이를 CAN 네트워크와 연결해 사용함으로써 시스템의 원가를 낮추고 있다. 또한 CAN의 경우 버스 속도가 최대 1Mbps라는 단점과 이벤트 트리거(event trigger) 기반의 전송방식 때문에 전송 시간의 예측이 불가능한 구조적인 문제점을 안고 있어서 최근에는 최고 10Mbps의 속도를 지원하며 TDMA 방식을 사용하여 시간 예측성을 확보한 FlexRay가 도입되어 광범위하게 적용되고 있다.

이와 같은 제어용 네트워크 이외에 차량 내부의 각종 음성, 영상을 처리하기 위한 인포테인먼트(Infotainment)용 네트워크가 있는데 이러한 용도로는 광케이블을 이용해 최대 150Mbps의 속도를 갖는 MOST, IEEE-1394를 변형해 최대 800Mbps의 속도를 갖는 IDB-1394, 기존 이더넷을 자동차용으로 변경한 차량용 이더넷 등이 본격적으로 적용 또는 검토되고 있다[2][3]. 이와는 별도로 ITS, ADAS, 자율주행, 커넥티드 카 등 차량 외부와의 V2X(Vehicle to X) 용도의 무선통신을 위한 네트워크로는 DSRC, WAVE, CALM 등의 규격들이 적극적으로 연구되고 있다[4].

차량용 네트워크와 관련된 또 다른 연구 분야는 통합 백본(backbone)과 게이트웨이(gateway)에 대한 연구이다. 통합 백본은 기존의 제어용 네트워크, 인포테인먼트용 네트워크 등을 하나로 묶어서 관리하는 통합 네트워크 시스템이며 높은 통신 속도와 TDMA 기반의 버스 관리가 필수사항인데 최근에는 MOST와 차량용 이더넷 기술을 백본으로 활용하기 위한 연구가 활발히 진행되고 있다[3]. 특히 차량용 이더넷의 경우는 브로드콤사의 BroadR-Reach 기술을 이용하여 하나의 트위스트 페어 케이블을 사용하여 100Mbps의 전송이 가능한 규격(100BASE-T1)이 이미 차량에 적용되었으며 1Gbps 속도를 위한 RTPGE(Reduced Twisted Pair Gigabit Ethernet) 기술 및 TDMA 기반의 TTEthernet(Time-Triggered Ethernet)에 대한 연구가 진행 중이다[3].

게이트웨이의 기본 목적은 차량 내부의 다양한 종류의 네트워크들을 연결해서 필요시 서로 다른 네트워크로 데이터를 안정적으로 변환하고 전달해 주는 것이며 주요한 기능은 메시지의 변경, 라우팅(routing), 실시간 처리, fault tolerance 등이 있는데 이 항목들은 기존의 많은 연구[5][6][7]에서 다루고 있기 때문에 본 논문에서는 상대적으로 연구가 적은 메시지의 효율적인 변환 및 전송 알고리즘에 대해서 중점적으로 검토하였다. 특히 지금까지의 연구에서는 하나의 네트워크에서 또 다른 하나의 네트워크로만 데이터를 변환해 전달하는 1:1 게이트웨이 기능만이 검토가 되었기 때문에 본 논문에서는 하나의 네트워크 입력을 복수개의 네트워크로 변환, 전송하는 1:n 게이트웨이용 알고리즘을 제안하였다. 본 알고리즘은 매핑 테이블을 사용하기 때문에 출력 네트워크로의 데이터 전송 시에 출력의 우선순위를 변경할 수 있으며 또한 네트워크간의 전송 데이터 포맷이 변경될 경우에도 소프트웨어의 변경이 없이 매핑 테이블 만을 변경함으로써 변경 내용을 쉽게 차량에 적용할 수 있는 추가적인 장점이 있다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문에서 다루는 CAN, FlexRay와 기존 게이트웨이에 대해서 간략히 설명을 실시하고 3장에서는 새롭게 제안하는 우선순위 기반 매핑 테이블 알고리즘의 구현을 위한 기본적인 사항 및 하드웨어와 소프트웨어의 구조를 제안하였고 4장에서는 우선순위 기반 매핑 테이블 및 이에 대한 구체적인 알고리즘을 제시하였다.

Ⅱ. 관련 기술 검토

본 장에서는 제안된 게이트웨이의 구현과 관련된 기술인 CAN, FlexRay와 게이트웨이와 관련된 기존의 연구에 대해서 간략한 설명을 실시한다.

2.1 CAN (Controller Area Network)

CAN은 1980년대에 독일의 자동차 부품회사인 보쉬(Bosch)사에서 개발하여 1993년 ISO 표준으로 제정된 자동차용 표준 네트워크이다. CAN은 시간과 관계없이 전송할 메시지가 발생할 때만 버스로 출력하는 이벤트 트리거 기반의 통신 프로토콜이다. 최대 1Mbps까지의 전송속도를 지원하며 여러 개의 노드(ECU)들이 한 쌍의 트위스트 페어 케이블을 사용하여 멀티드롭 형태로 버스가 구성된다. 각 노드들은 복수개의 송신 메시지들을 가질 수 있는데 하나의 메시지는 고유의 메시지 ID를 갖게 되며 메시지 ID의 우선순위에 따라 버스 사용 우선권을 갖는 CSMA/CA 방식으로 버스를 관리한다. 메시지 ID의 숫자가 낮은

것이 버스 사용의 우선권을 갖는다.

CAN 규격에는 ID가 11비트인 표준 데이터 프레임 포맷(그림 1)과 ID가 29비트인 확장 데이터 프레임 포맷이 있다(그림 2)[8]. 모든 CAN 메시지는 프레임 단위로 구분되며 프레임은 여러 개의 필드로 구분되고 DLC(Data Length Code) 필드에 해당 프레임의 데이터 바이트 수를 지정하여 최대 8바이트까지의 데이터를 가변적으로 전송할 수 있다. 그림 1, 그림 2에서 보이듯이 하나의 CAN 메시지는 DLC에 따라 데이터의 수가 정해지며, 6개의 동일레벨 출력을 허용하지 않는 bit stuffing 기법을 사용하기 때문에 프레임 길이는 식 1, 식 2와 같이 가변적으로 표현된다.

$$\text{표준 프레임 길이} = 47bit + (8 \times DLC) + \frac{34 + (8 \times DLC)}{5} \quad (1)$$

$$\text{확장 프레임 길이} = 67bit + (8 \times DLC) + \frac{54 + (8 \times DLC)}{5} \quad (2)$$

CAN 노드에서의 수신은 특정 노드로부터 전송이 시작되면 버스에 연결된 모든 노드의 CAN 컨트롤러들은 메시지 ID를 기반으로 설정되는 acceptance filter를 이용해서 수신여부를 판단하여 수신이 허용된 ID의 메시지만 수신한다.

비록 CAN은 낮은 전송속도와 이벤트 트리거 기반의 전송방식에서 발생하는 구조적인 문제점을 가지고 있으나 근래에는 CAN 컨트롤러를 내장하는 다양한 마이크로컨트롤러(MCU)들이 많은 반도체 업체에서

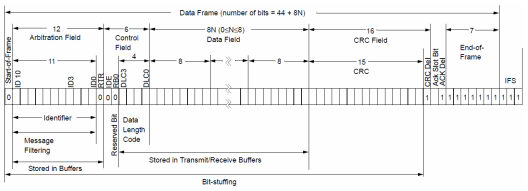

그림 1. CAN의 표준 데이터 프레임 포맷(CAN 2.0A)
Fig. 1. CAN standard data frame format

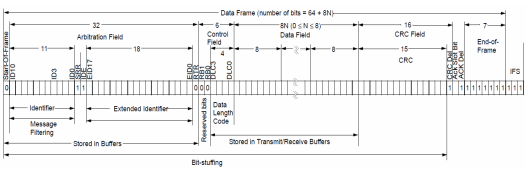

그림 2. CAN의 확장 데이터 프레임 포맷(CAN 2.0B)
Fig. 2. CAN extended data frame format

생산되고 있기 때문에 낮은 가격과 부품선택의 폭이 넓어져서 현재는 자동차뿐 아니라 다양한 산업분야에서 폭넓게 사용되어지고 있다.

CAN의 전송속도 문제점을 개선하기 위해 최고 8Mbps로 최대 64바이트의 데이터 전송이 가능한 CAN FD(CAN with a Flexible Data-rate)를 2012년 보쉬사가 제시했으나 아직은 적용이 되지 않고 있으며, 전송시간을 보장하기 위하여 TDMA 방식의 TTCAN(Time Triggered CAN)이 제안되어 2004년 ISO로 표준화 되었으나 이 방식 역시 전송속도 문제로 널리 사용되지 않고 있으며 기존의 CAN을 소프트웨어적인 방법으로 전송시간을 보장하기 위한 연구도 일부 진행되고 있다[9].

2.2 FlexRay

CAN은 기존의 자동차 제어부문에서 폭넓게 사용되고 있었으나 brake-by-wire, steer-by-wire 등의 x-by-wire 제어 시스템의 등장은 기존의 제어 방식보다 훨씬 더 빠르고 정확한 시간에 제어가 되어야하기 때문에 CAN을 이러한 분야에 적용하기에는 불가능하게 되었다. 따라서 x-by-wire 등의 고속 제어에 적합한 통신 프로토콜을 개발하기 위해 2000년에 주요 완성차 업체들, 부품 업체들이 FlexRay 컨소시엄을 구성해 본격적인 개발을 시작했고 2009년도에 ISO로 표준화 되었다. FlexRay는 전송시간의 예측성을 확보하기 위해 TDMA 기반의 통신을 하며 채널당 최대 10Mbps의 전송속도를 지원하고 필요시 두 개의 채널로 동일한 데이터를 보내 한 채널에서 문제가 발생해도 또 다른 채널로 올바른 데이터를 확보할 수 있는 fault tolerance 특성을 갖는 것이 가장 큰 특징이다.

그림 3[10]은 FlexRay의 사이클에 대한 구성이며 하나의 통신 사이클은 반복적으로 순환되고 시스템 설계 시에 사이클 시간 및 세부사항들이 고정된 값으로 결정된다. 하나의 사이클은 일반적으로 정적 세그먼트(Static Segment), 동적 세그먼트(Dynamic Segment), 심볼 윈도우(Symbol Window), 네트워크 유휴 시간(Network Idle Time)으로 구성되며 심볼 윈도우는 네

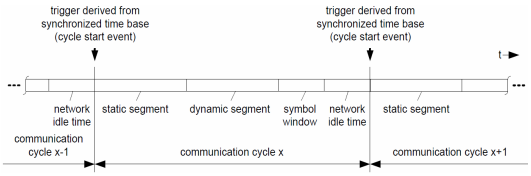

그림 3. FlexRay 통신의 사이클
Fig. 3. Basic communication cycle

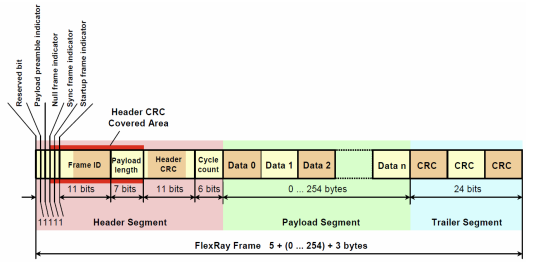

그림 4. FlexRay의 프레임 포맷
Fig. 4. FlexRay frame format

트워크의 유지와 시작, 네트워크 유휴 시간은 노드간의 클록을 동기화하는 용도로 사용된다. 그림 4[10]는 정적 세그먼트와 동적 세그먼트에서 공통으로 사용하는 프레임 포맷이며 하나의 프레임에는 최대 254바이트까지의 데이터를 전송할 수 있다.

정적 세그먼트 구간(그림 5[10])은 동일한 시간 길이를 갖는 슬롯(slot)들로 구성되고 각 슬롯 구간마다 시스템 설계 시에 정해진 노드들이 출력할 데이터의 유무에 상관없이 무조건 버스의 사용 권리를 갖는다. 각 노드들은 주어진 슬롯번호에 맞추어 전송을 보장 받기 때문에 시간 예측이 가능한 제어시스템을 구성할 수 있다. 출력 노드는 프레임 ID 필드에 자신의 슬롯 번호인 ID를 포함해 전송을 하여 다른 노드들이 프레임의 수신여부를 판단하게 한다. 특정 슬롯의 출력이 완료되면 슬롯번호는 1이 증가되어 다음번 슬롯번호를 할당받은 노드에게 버스의 사용권을 넘겨주는데 하나의 사이클 시간동안에 하나의 노드가 복수개의 슬롯들을 사용하도록 시스템을 설계할 수도 있다. 하나의 프레임에 최대 254바이트까지의 데이터를 전송할 수 있으나 모든 슬롯은 동일한 시간 크기를 가져야 하기 때문에 시스템 설계 시에 전체 시스템의 효율성을 위해 가장 긴 길이의 페이로드(데이터)를 기준으로 슬롯의 크기를 결정한다.

동적 세그먼트 구간(그림 6[10])은 항상 일정한 슬롯 시간을 갖는 정적 세그먼트에 할당하기에는 비효율적인 작은 크기의 데이터 또는 이벤트 메시지 등의 비주

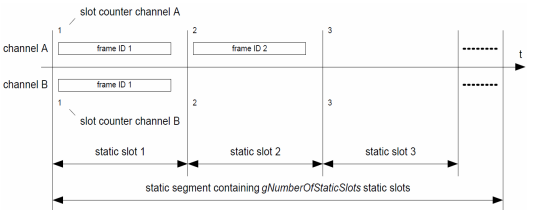

그림 5. FlexRay의 정적 세그먼트
Fig. 5. FlexRay Static Segment

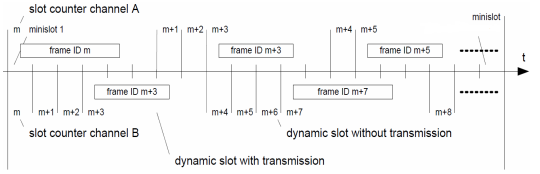

그림 6. FlexRay의 동적 세그먼트
Fig. 6. FlexRay Dynamic Segment

기적인 데이터를 전송하는 구간으로 사용되는데 시스템 설계 시에 결정되는 x개의 짧은 시간을 갖는 미니슬롯(mini-slot)들로 구성이 된다. n개의 슬롯으로 구성된 정적 세그먼트가 끝나게 되면 슬롯번호는 n+1이 되는데 동적 구간에서도 슬롯번호를 기준으로 할당된 노드가 버스를 사용한다. 그러나 정적 세그먼트와의 차이점은 버스 사용권을 부여받은 노드가 송신할 데이터가 없을 경우에는 매우 짧은 미니슬롯 시간이 경과한 후 자동으로 슬롯번호가 증가되어 다음 슬롯번호에 할당된 노드가 데이터를 출력할 수 있도록 한다. 만약 모든 노드들이 x개의 미니슬롯이 경과해도 전송을 하지 않아 슬롯번호가 n+x를 초과할 경우에는 동적 세그먼트가 종료된다. 버스를 할당받은 특정 노드가 출력을 할 경우에는 그림 4의 포맷으로 전송하는데 정적 세그먼트와는 다르게 payload length 필드에 설정한 길이만큼의 데이터만을 전송하고 즉시 버스 사용권을 반납하는 구조로 되어있기 때문에 프레임 길이는 가변적이 된다. 이러한 이유로 동적 세그먼트의 전송방식을 이벤트 트리거 기반 FTDMA(Flexible TDMA)라고 한다. 정적, 동적 세그먼트에서 사용되는 데이터 포맷은 6비트의 사이클 번호를 갖는데 하나의 사이클이 완료될 때마다 번호가 1씩 증가하며 0부터 63까지를 반복하게 된다. 이를 이용해 동일 슬롯번호에서 출력된 메시지도 사이클 번호에 따라서 내용이 다른 데이터를 출력할 수 있는 융통성을 갖는다.

FlexRay가 최초로 상용화된 차량은 2007 BMW X5 SAV 모델로서 전자식 Damper 제어 시스템에 사용되었으며 이 시스템은 5개의 노드를 스타 구조로 연결하고 10Mbps 버스 속도, 정적 세그먼트 3ms, 동적 세그먼트 2ms, 최대 16바이트의 페이로드로 구성되어 있으며 정적 세그먼트는 2.5ms, 5ms, 10ms, 20ms, 40ms 주기의 task용 데이터 통신을 위해 사용되었다[11]. 이렇듯 FlexRay는 x-by-wire 시스템에 잘 적응되는 구조를 가지고 있어서 많은 장점이 있으나 CAN과 대비해 높은 가격 때문에 많은 자동차 업체들은 전면적인 적용보다는 일반 제어에는 비교적 저가인 CAN을 사용하고 x-by-wire 제어에는 FlexRay를

적용하여 두 네트워크를 게이트웨이로 연결하여 사용하고 있다.

2.3 기존 게이트웨이 관련 연구

게이트웨이와 관련된 기존의 연구들은 CAN-FlexRay와 관련된 연구[5], LIN-CAN-FlexRay와 관련된 연구[6] 그리고 CAN-FlexRay-Ethernet과 관련된 연구[7] 등이 있다. 대부분의 연구들은 특정 네트워크에서의 입력을 또 다른 하나의 네트워크로만 변환, 전송하는 1:1 게이트웨이 기법에 대해서 다루고 있으나 실제의 게이트웨이 시스템에서는 하나의 입력에서 다양한 네트워크로 데이터를 변환, 전송하여 정보를 광범위하게 공유하는 1:n 게이트웨이 변환기능이 필요하다.

또한 CAN과 같은 이벤트 기반의 네트워크로 데이터를 전송할 때에도 기존의 CAN 자체의 ID 기반 우선순위 이외에 전체 게이트웨이 네트워크 시스템 입장에서의 우선순위가 새롭게 정해질 필요가 있으며, 네트워크간의 전송 데이터 포맷이 변경될 경우에도 소프트웨어의 변경이 없이 쉽게 차량에 적용할 수 있는 구조에 대한 연구가 필요하다.

따라서 본 논문에서는 이러한 기본의 연구에서 부족했던 문제점들을 개선할 수 있는 매핑 테이블 기반의 새로운 게이트웨이 알고리즘에 대한 제안을 하였다.

Ⅲ. 게이트웨이의 기본 설계

본 장에서는 제안하는 알고리즘을 구현하기 위한 게이트웨이의 데이터 변환 방법, 게이트웨이용 네트워크 시스템의 구성을 설명하고 이를 실제 구현하기 위한 게이트웨이의 하드웨어, 소프트웨어 구조를 제시하였다.

3.1 데이터 변환의 종류

게이트웨이에서 처리되는 데이터는 기본적으로 가장 낮은 단계인 신호(signal)와 하나 또는 복수개의 신호로 구성되는 PDU(Protocol Data Unit), 하나 또는 복수개의 PDU와 해당 프로토콜용 ID, payload 등의 header와 CRC 등의 trailer로 구성되는 프레임으로 구분할 수 있으며 따라서 게이트웨이에서 변환되는 데이터의 변환 형태는 프레임 단위 변환과 신호 단위 변환으로 크게 구분할 수 있다.

프레임 단위 변환의 경우는 입력된 프레임의 PDU를 그대로 유지하면서 목적지 네트워크에서 사용하는 프레임에 맞게 header와 trailer 정보를 변경해 프레임을 재구성하여 전송하는 방식이며 주로 동일한 특성을 갖는 네트워크(homogeneous network) 사이에서의 변환에 적합한 변환기법이며 신호 단위 변환은 이종 네트워크(heterogeneous network) 사이의 변환에 적합한 방식인데 입력되는 PDU에서 필요한 신호들만을 추출, 변환하여 출력하는 방법이다. OSEK/VDX에서는 프레임 단위의 변환은 PDU Router, 신호 단위의 변환은 COM 모듈을 통해서 처리하나 본 논문에서는 이를 구분하지 않고 하나의 신호 단위 변환 테이블 만을 이용해 간결하고 일관된 변환작업을 실시하였다.

3.2 게이트웨이의 구성

본 논문에서는 비교적 작은 크기를 갖는 많은 종류의 실시간 데이터들이 처리되는 차량 제어용 네트워크에 대해서 중점적으로 다루고 있다. 따라서 차량 제어용과는 직접 연관되지 않는 인포테인먼트용 네트워크와 본 논문에서 제안한 알고리즘이 쉽게 적용될 수 있는 백본 네트워크의 경우는 시스템 구성에서 제외하였다.

그림 7은 본 논문에서 검토하는 게이트웨이의 구성도이며 게이트웨이를 중심으로 두 개의 FlexRay 버스, HS(High Speed) CAN 버스, LS(Low Speed) CAN 버스를 각각 구성하였다. HS CAN(ISO11898-2)은 최대 1Mbps의 고속 CAN으로 주로 파워트레인 제어용도로 사용되며, LS CAN(ISO11898-3)은 최대 125Kbps를 지원하는 저속 CAN으로 주로 도어나 시트 등의 저속으로 제어가 가능한 부문에 사용된다. HS CAN 버스에는 일반 제어용 노드 이외에 차량 진단용 OBD-II(On Board Diagnostic-II)가 연결된다. 또한 V2X 관련한 WAVE의 경우도 별도의 WAVE 모듈에서 이를 처리하고 차량 내부와의 데이터 송수신은 CAN 버스를 사용하는 방법이 연구[12]되고 있기 때문에 WAVE 모듈도 HS CAN 버스에 연결하는 것으로 구성하였다. LIN 네트워크의 경우도 특정 CAN 노드가 마스터 기능을 수행하면서 LIN 통신 전체를 제

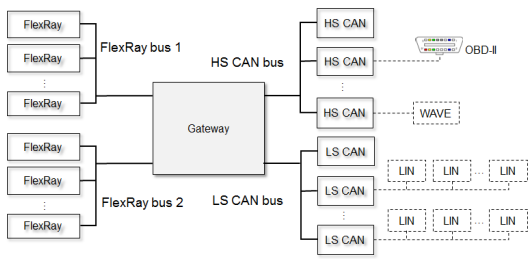

그림 7. 제안된 게이트웨이 시스템의 구성
Fig. 7. Configuration of gateway system

어하기 때문에 게이트웨이 입장에서는 단순히 LS CAN의 일부로 취급할 수 있다. FlexRay는 수 ms 주기를 갖는 x-by-wire 시스템의 실시간 제어에 활용된다.

3.3 하드웨어 구조

본 논문에서 검토되고 있는 게이트웨이는 그림 8의 하드웨어 구조를 가진다. 근래의 MCU들은 CAN, FlexRay 컨트롤러들을 칩에 내장하기 때문에 MCU의 외부에 신호변환용 트랜시버(transceiver)만 달아주면 쉽게 하드웨어를 구성할 수 있다. 이러한 일체형 MCU는 Freescale, ST, Infineon 등의 많은 자동차용 반도체 업체에서 생산하고 있으며 그림 8은 Freescale(NXP)사의 MCU와 트랜시버를 이용한 게이트웨이의 구성 예이다. 그림 9는 본 게이트웨이에서 사용되는 주요 인터럽트들의 우선순위 구성이다. FlexRay는 지정된 슬롯번호 시간에만 출력이 가능하기 때문에 FlexRay 출력과 관련되는 FlexRay PDU assemble 인터럽트들은 가장 높은 우선순위를 갖으며 시스템 타이머에 의해 주기적으로 발생된다. 4종류의 수신 인터럽트들(Rx interrupt)은 낮은 우선순위를 갖는데 이는 개별 컨트롤러들이 복수개의 내부 수신버퍼를 가지고 있기 때문에 다소 지연된 처리가 가능하기 때문이다.

본 논문에서 제시되는 모든 입출력 매핑 테이블들과 task 관련 프로그램들은 그림 8의 Flash ROM영역

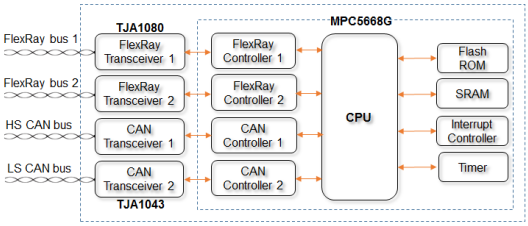

그림 8. 게이트웨이 하드웨어 구성
Fig. 8. Gateway hardware architecture

Interrupt name	Priority	Source	Processing
FlexRay 1 Rx interrupt	Low	FlexRay 1 controller	Invoke FR1_Rx task
FlexRay 2 Rx interrupt	Low	FlexRay 2 controller	Invoke FR2_Rx task
HS CAN Rx interrupt	Low	HS CAN controller	Invoke HSCAN_Rx task
LS CAN Rx interrupt	Low	LS CAN controller	Invoke LSCAN_Rx task
HS CAN Tx interrupt	Mid	HS CAN controller	Invoke HSCAN_Tx task
LS CAN Tx interrupt	Mid	LS CAN controller	Invoke LSCAN_Tx task
FlexRay 1 PDU assemble	High	System Timer-1	Invoke FR1_Tx task
FlexRay 2 PDU assemble	High	System Timer-2	Invoke FR2_Tx task

그림 9. 인터럽트 구성
Fig. 9. Interrupt configuration

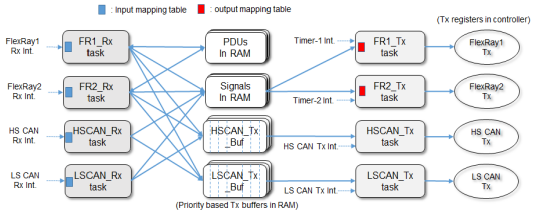

그림 10. 소프트웨어 구조
Fig. 10. Software architecture

에 저장하여 실행되게 함으로써 테이블의 변경 등이 필요한 경우에도 쉽게 변경이 가능한 구조를 갖는다. 또한 SRAM의 영역이 충분히 확보될 경우에는 Flash ROM에 저장된 테이블들과 프로그램들을 초기 전원 공급 시 SRAM으로 복사하여 SRAM에서 동작하게 함으로써 보다 향상된 처리속도를 얻을 수 있다.

3.4 소프트웨어 구조

그림 10은 본 논문에서 제안하는 알고리즘을 위해 구성되는 소프트웨어 task들, 데이터의 흐름 및 task의 호출 등을 정리한 내용이다. 게이트웨이의 소프트웨어는 총 4개의 입력변환용 task(*xxx*_Rx), 타임 트리거 기반의 FlexRay 출력용 task 2개(FR*x*_Tx) 그리고 이벤트 트리거 기반의 CAN 출력용 task 2개(*x*SCAN_Tx)로 구성되어 있다.

3.4.1 입력 처리용 소프트웨어 구조

4개의 Rx task는 해당 버스 컨트롤러에서 생성하는 데이터 수신 인터럽트에 의해 불려진다. 이 task들은 각각 4.1절에서 제안되는 입력용 매핑 테이블을 이용하여 입력된 데이터를 출력 신호들로 변환한 후 FlexRay용 출력 신호들은 테이블에서 지정하는 특정 RAM 위치에 데이터를 저장하고, CAN용 출력 신호들은 테이블에서 지정하는 우선순위에 맞추어 다단계 우선순위 출력버퍼(*x*SCAN_Tx_Buf)에 모든 신호들을 저장한다.

다단계 우선순위 출력버퍼는 하나의 단계에도 복수개의 데이터를 저장할 수 있는 구조로 만들고 이를 큐(queue) 기법으로 관리하여 출력의 지연 시에도 데이터의 손실을 방지한다. 주기가 빠른 FlexRay 입력의 경우는 이전 사이클의 PDU를 RAM에 보관하여 변경된 데이터만을 변환, 전송하여 전체적인 게이트웨이 시스템의 효율성을 높인다.

3.4.2 출력 처리용 소프트웨어 구조

FlexRay 버스로의 출력은 게이트웨이에게 할당된

특정 슬롯번호 시간에만 가능하기 때문에 시스템 타이머 인터럽트를 이용해서 주어지는 슬롯시간 이전에 개별신호에서 PDU로의 변환작업이 가능한 시간적 여유를 가지고 주기적으로 각각의 출력 task(FR*x*_Tx)를 수행시키는데 출력 task에서는 4.2절의 출력용 매핑 테이블을 이용하여 RAM에 흩어져 있는 필요 신호들을 모아서 이를 PDU로 조합하여 해당 FlexRay 컨트롤러의 출력 레지스터에 미리 저장함으로써 주어진 슬롯번호 기간에 맞추어 자동으로 출력되게 한다.

CAN 버스로의 출력은 FlexRay와는 다르게 이벤트 방식으로 진행되는데 입력변환 task에서는 변환된 모든 신호들을 해당 CAN용 다단계 우선순위 출력버퍼에 저장한 후 즉시 특정 CAN용 출력 task(*x*SCAN_Tx)를 불러서 출력을 진행한다. CAN용 출력 task는 호출될 때마다 항상 다단계 출력버퍼 중에서 가장 우선순위가 높은 버퍼의 내용을 먼저 처리하는데 만약 상위 우선순위 버퍼에 출력할 내용이 없을 때는 다음 단계의 우선순위 출력버퍼의 내용을 CAN 컨트롤러의 출력 레지스터로 옮겨주며 자신의 task를 종료하게 된다. 이후 CAN 컨트롤러에서는 CAN 출력 task에서 지정한 데이터의 출력이 완료되면 출력 완료 인터럽트(transmission completed interrupt)를 발생시키고 이 인터럽트에서 다시 CAN 출력 task를 호출하여 우선순위 버퍼의 다음 내용을 출력할 수 있게 하는데 이러한 절차는 다단계 출력버퍼의 모든 내용이 출력 완료될 때까지 반복하게 된다.

Ⅳ. 알고리즘의 설계 및 검증

매핑 테이블의 종류는 버스 당 하나씩의 입력용 매핑 테이블과 FlexRay와 같은 타임 트리거 기반 버스에서 사용하는 출력용 매핑 테이블로 나누어진다. 따라서 본 논문에서의 매핑 테이블은 그림 10에서와 같이 4개의 입력용 매핑 테이블과 2개의 FlexRay 출력용 매핑 테이블로 구성된다.

4.1 입력용 매핑 테이블 구조 및 변환 알고리즘

그림 11은 FlexRay 버스1의 입력 변환 task의 흐름도이며 그림 12는 여기서 사용하는 입력용 매핑 테이블의 구조이다. FlexRay 버스1의 수신부에 데이터가 입력되면 해당 컨트롤러의 수신 인터럽트는 FlexRay 버스1용 입력변환 task를 호출하게 되는데 이 task에서는 입력된 PDU 정보를 기반으로 입력용 매핑 테이블을 이용하여 신호 단위로의 변환을 한다. 입력된 데이터는 우선적으로 그림 12.a의 마스터 테이블의 사이

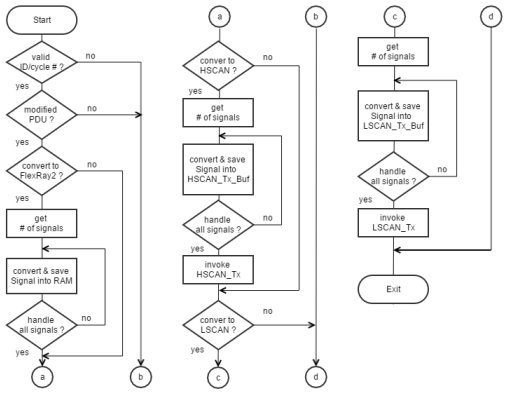

그림 11. FR1_Rx task의 흐름도
Fig. 11. FR1_Rx task Flow Chart

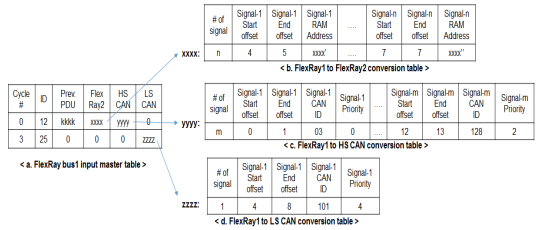

그림 12. FlexRay 입력 처리용 매핑 테이블 구조
Fig. 12. Mapping table for FlexRay input

클 번호, ID를 입력된 데이터의 사이클 번호, ID와 비교하여 테이블에 일치되는 정보가 없을 시, 또는 일치하더라도 RAM kkkk번지에 저장된 이전 PDU와 동일한 내용인 경우에는 재전송할 필요가 없으므로 변환작업을 종료하며 PDU의 변경이 있을 경우에만 해당 줄(인덱스)의 정보를 이용하여 FlexRay 버스2, HS CAN, LS CAN 버스로의 전송여부를 차례로 판단하게 된다.

사이클 번호 0, ID 12에서 입력된 데이터는 그림 12.a에서 보이듯이 FlexRay 버스2, HS CAN 항목은 각각의 변환 테이블의 위치가 xxxx, yyyy로 지정되어 있으므로 이 주소를 이용해 변환과 전송을 할 필요가 있음을 의미하고, LS CAN 항목에는 0(null)이 있으므로 재전송할 필요가 없음을 의미한다. FlexRay 버스2로의 변환과 출력에 사용되는 xxxx 번지의 변환 테이블(그림 12.b)은 해당 PDU가 몇 개의 세부 신호들로 나누어지는지와 분해될 각 신호들의 PDU내에서의 위치(시작 위치, 종료 위치) 그리고 변환된 신호가 저장될 게이트웨이의 RAM 주소가 분해될 신호의 수(n)만큼 반복하여 표에 기록된다. 변환 표에 의해 FlexRay 버스2용 변환이 종료되어 모든 신호들이 RAM에 저장되면 이번에는 HS CAN용 데이터 변환

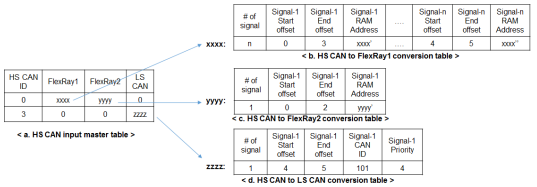

그림 13. HS CAN 입력 처리용 매핑 테이블 구조
Fig. 13. Mapping table for HS CAN input

을 실시하는데 yyyy 번지의 변환 테이블(그림 12.c)을 이용하게 된다. yyyy번지의 변환 테이블의 경우 나누어질 CAN 메시지의 개수(m), PDU에서의 시작과 종료 위치, 변환될 CAN ID 및 HS CAN 버스로 출력시의 우선순위 정보를 포함하는데 이를 이용해 변환을 실시하고 변환된 내용을 HS CAN의 우선순위 기반 다단계 출력버퍼에 저장하는 것을 지정된 신호의 수(m)만큼 반복한다. 모든 저장이 완료되면 HS CAN용 출력 task를 호출하여 CAN 버스로 데이터를 출력하게 한다. HS CAN용 변환작업이 종료되면 동일한 방법으로 LS CAN용 변환작업을 실시하는데 본 예제에서는 마스터 테이블(그림 12.a)에 0으로 채워져 있기 때문에 LS CAN용 변환은 생략된다.

그림 13은 HS CAN 입력 변환 task에서 사용하는 매핑 테이블의 구조이다. FlexRay의 동일한 방법으로 다른 버스로의 변환 여부를 입력된 CAN ID와 입력용 마스터 매핑 테이블(그림 13.a)과 비교하여 판단하고 필요시 우측의 변환 테이블들을 이용해 신호들을 변환하여 LS CAN 출력일 경우 변환 테이블에서 지정한 우선순위에 맞는 다단계 출력버퍼에 저장하고 FlexRay용 출력일 경우 테이블에서 지정한 RAM 위치에 저장하게 된다.

FlexRay 버스2, LS CAN 입력용 처리방식은 각각 FlexRay 버스1과 HS CAN에서 설명된 내용과 동일한 방법으로 구성된다.

4.2 출력용 매핑 테이블 구조 및 변환 알고리즘

그림 14는 타이머 인터럽트에 의해 주기적으로 호출되는 FlexRay 출력 task에서 신호들의 조합을 목적으로 사용하는 출력용 매핑 테이블의 구조이다. FlexRay용 출력 task도 출력 마스터 매핑 테이블(그림 14.a)에서 자신의 상황에 맞는 사이클 번호와 ID를 찾고 이를 이용해 처리할 정보가 포함된 조합표(그림 14.b)의 주소를 찾아서 FlexRay 출력용 PDU를 만들게 된다. 조합표에는 PDU를 구성하는 신호들의 PDU 내에서의 삽입될 위치 정보 및 삽입될 신호들이 저장된 RAM의 주소가 정리되어 있다. 이 정보를 통해 만

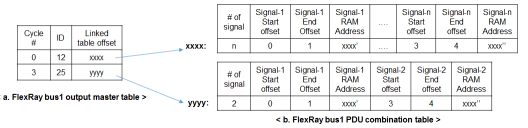

그림 14. FlexRay 출력 처리용 매핑 테이블 구조
Fig. 14. Mapping table for FlexRay output

들어진 PDU는 FlexRay 컨트롤러의 출력 레지스터에 저장되어 FlexRay 컨트롤러가 주어진 슬롯번호가 되면 header와 trailer를 조합해 완성된 프레임 포맷으로 출력할 수 있게 한다.

FlexRay 버스1, 2의 출력 처리용 매핑 테이블의 값은 서로 다르지만 동일한 알고리즘으로 처리되며 CAN의 경우는 이벤트 기반의 처리이기 때문에 별도의 출력용 매핑 테이블은 존재하지 않는다.

만약 FlexRay나 CAN 이외의 다른 네트워크 방식이 적용될 경우라도 이벤트 트리거 기반의 네트워크인 경우는 CAN과 동일한 개념으로, 타임 트리거 기반의 네트워크인 경우는 FlexRay와 동일한 개념으로 출력을 설계하면 쉽게 적용이 가능하다.

4.3 알고리즘의 검증

게이트웨이의 가장 핵심적인 기능은 입력된 메시지를 변환하고 변환된 데이터 포맷을 해당 목적지 네트워크로 빠른 시간 내에 전달하는 것이다. 이중에서도 입력 메시지의 변환여부에 대한 판단 및 변환정보를 얻기 위한 입력 테이블(그림 12.a, 그림 13.a)의 검색시간이 매우 중요한데 본 논문에서는 새롭게 제시한 1:n 게이트웨이의 변환 알고리즘과 기존의 1:1 게이트웨이의 변환 알고리즘에서 입력 테이블 검색시간에 대한 시뮬레이션을 실시하였다.

입력 테이블 검색시간과 관련된 시뮬레이션은 일반적으로 많이 사용되고 있는 ARM Cortex-A7 쿼드코어(900Mhz)를 이용하였고 검색 프로그램은 RAM상에서 실행하였으며 전체 시스템의 구성은 그림 7에서와 같이 4개의 네트워크를 하나의 게이트웨이에 연결하는 것으로 설정하였다.

실험조건으로 모든 입력 테이블들이 각각 25개, 50개, 100개 그리고 200개의 인덱스(메시지 ID의 종류)를 동등하게 갖는 것으로 가정하였다. 검색 이후 메시지를 신호로 변경하고 이를 출력하는 방법은 1:1 변환방식이나 1:n 변환방식이 서로 유사하므로 이 부분에 대한 시뮬레이션은 생략하였다.

그림 15는 시뮬레이션의 결과를 정리한 그림이며 그림에서 볼 수 있듯이 기존 1:1 변환방식의 경우 3번

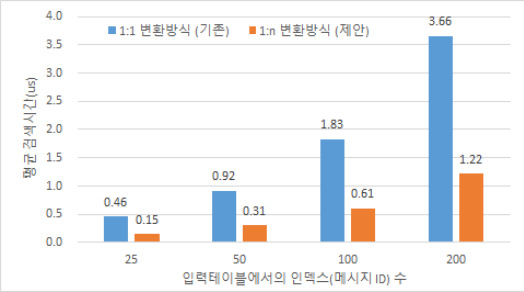

그림 15. 시뮬레이션 결과
Fig. 15. Simulation result

의 입력 테이블 검색이 필요하나 1:n 변환방식의 경우는 한 번의 입력 테이블 검색으로 모든 변환작업에 대한 정보들을 검색할 수 있기 때문에 1:n 변환방식의 검색시간이 1:1 변환방식과 비교하면 ⅓로 단축됨을 알 수 있었다. 또한 시뮬레이션 결과를 통해서 게이트웨이 시스템을 설계할 때 각 입력 테이블에서의 인덱스(메시지 ID 종류)의 수를 최적화하는 것이 매우 필요함을 알 수 있었다.

V. 결 론

본 논문에서는 우선순위 매핑 테이블을 기반으로 한 차량 제어 네트워크용 게이트웨이에 대한 알고리즘 및 기본 설계를 제시하였다.

제시된 알고리즘은 다양한 종류의 네트워크가 연결된 게이트웨이에서 1:n 네트워크로의 변환 및 전송에 대한 구체적인 방식을 제안하였으며 또한 CAN 네트워크로 복수 데이터를 출력할 때도 기존의 FIFO방식이 아닌 게이트웨이 시스템의 설계 시에 부여하는 새로운 우선순위에 의거해 출력순서를 바꿀 수 있는 구조를 가지고 있다. 아울러 모든 데이터의 변환, 출력과 관련된 처리과정은 표준화된 매핑 테이블의 구조를 도입하였기 때문에 변환되는 데이터의 구조 변경이 필요할 시 모든 프로그램을 변경하는 것이 아니라 단지 매핑 테이블들만을 변경함으로써 간단히 적용될 수 있다는 장점도 가지고 있다. 또한 이벤트 트리거 기반인 CAN과 타임 트리거 기반인 FlexRay 네트워크에 대한 개별적인 입력, 변환, 출력 알고리즘을 명확히 제시하고 있기 때문에 향후 다른 종류의 네트워크가 추가될 경우에도 본 알고리즘을 응용하여 쉽게 적용이 가능하다.

본 논문에서는 차량 제어용 게이트웨이와 관련하여 우선순위 매핑 테이블의 개념 및 이를 처리하기 위한 알고리즘에 대한 제안 및 기본적인 하드웨어, 소프트웨어 구조에 대하여 다루었다.

향후에는 본 논문에서 제시된 알고리즘과 기본 설계 내용들을 기반으로 실제의 시스템을 구현하고 평가하는 연구가 추가로 필요하며 특히 최근의 표준화 동향과 발맞추어 OSEK/VDX OS, AUTOSAR와 연동된 구현이 필요할 것으로 판단된다.

References

[1] G. Leen and D. Hefferman, “Digital Networks in the automotive vehicle,” *IEEE Computer and Control Eng. J.*, vol. 10, no. 6, pp. 257-266, Dec. 1999.

[2] S. Tuohy, M. Glavin, and E. Jones, “Next generation wired intra-vehicle networks, A review,” in *IEEE Intell. Veh. Symp.(IV)*, pp. 777-782, Gold Coast, Australia, Jun. 2013.

[3] Charles M. Kozierok, C. Correa, Robert B. Boatright, and J. Quesnelle, *Automotive Ethernet: The Definitive Guide*, Intrepid Control Systems, 2014.

[4] H.-S. Oh, “V2X communication technology, recent trends and practical issues,” *J. KICS*, vol. 30, no. 11, pp. 3-7, Oct. 2010.

[5] Y. G. Bae, M. H. Kim, S. Lee, and K. C. Lee, “Implementation of node mapping-based FlexRay-CAN gateway for in-vehicle networking system,” *Trans. Korean Soc. Automotive Eng.*, vol. 19, no. 6, pp. 37-45, Nov. 2011.

[6] S.-H. Seo, J. H. Kim, S.-H. Hwang, K. H. Kwon, and J. W. Jeon, “A reliable gateway for in-vehicle networks based on LIN, CAN, and FlexRay,” *ACM Trans. Embedded Comput. Syst.*, vol. 11, no. 1, Mar. 2012.

[7] J. H. Kim, S.-H. Seo, and N.-T. Hai, “A gateway framework for in-vehicle networks based on CAN, FlexRay, and Ethernet,” *IEEE Trans. Veh. Technol.*, vol. 64, no. 10, pp. 4472-4486, Oct. 2015.

[8] Microchip, MCP2515 data sheet, 2007.

[9] Sung-won Park, In-sung Kim, Dongik Lee, “Implementation of IEEE1588 for Clock Synchronization”, *The Journal of The Korean Institute of Communication Sciences(J-KICS)*

Vol.39 No.2, pp. 123-132, 2014.2.

[10] FlexRay Consortium, *FlexRay Communications System Protocol Specification Version 3.0.1*, Oct. 2010.

[11] A. Schedl, "Goals and architecture of FlexRay at BMW," *Vector FlexRay Symp. Stuttgart*, Mar. 2007.

[12] J. Kim, H. Seo, and S. Lee, "OSEK OS based gateway for interconnecting WAVE and CAN," *J. KICS*, vol. 39, no. 2, pp. 133-141, Feb. 2012.

김 의 룡 (Eui-Ryong Kim)

2010년 2월 : 한국산업기술대학교 컴퓨터공학과(공학사)
2012년 2월 : 한국산업기술대학교 IT융합학과(공학석사)
2016년 2월 : 한국산업기술대학교 컴퓨터공학과(공학박사)
<관심분야> 소프트웨어공학, 정보통신시스템, 객체지향

오 세 춘 (Se-Chun Oh)

1984년 2월 : 고려대학교 전자공학과 졸업(공학사)
1989년~2010년 : 삼성자동차, 삼성전자 근무
2015년 3월~현재 : 한국산업기술대학교 컴퓨터공학과 석사과정
<관심분야> 통신공학, 자동차공학, 디스플레이공학

김 영 곤 (Young-Gon Kim)

1983년 2월 : 경북대학교 전자공학과(공학사)
1985년 2월 : 연세대학교 본대학원 전자공학과(공학석사)
2000년 2월 : 한국과학기술원 전산학과(공학박사)
1985년~2007년 : KT 수석연구원
2007년~현재 : 한국산업기술대학교 컴퓨터공학과 교수
<관심분야> 소프트웨어공학, 정보통신시스템, 객체지향 분석 및 설계