DOI: 10.3745/KIPSTC.2008.15-C.3.199

CAN기반 시스템의 통신 신뢰성 검증

김 종 현[†] · 정 기 현[††] · 최 경 희[†††]

요 약

CAN은 처음에는 차량 네트워크에서의 사용을 위해 고안되었으나 잡음에 강하고 신뢰성이 높아 공장자동화 등에서도 많이 사용되고 있다. 그러나 1:1통신에서 네트워크 기반의 통신방식으로 변화되면서 각 장치의 기계적인 동작뿐만 아니라 전자, 소프트웨어적인 동작에 대해서도 철저한 검증이 필요하게 되었다. 본 논문은 CAN 기반 시스템에서 통신의 신뢰성을 데이터의 정확성에 대한 부분과 시간상의 정확성에 대한 부분을 검증하는 방법을 제시하고 있다.

키워드 : 임베디드 시스템, CAN, 신뢰성, 차량용 네트워크

A Study on CAN Based System Reliability Test

Jong-Hyun Kim[†] · Ki-Hyun Chung[††] · Kyung-Hee Choi[†††]

ABSTRACT

Controller Area Network was developed originally for in-vehicle communication network. But it is now widely used for factory automation because of its properties such as strong noise resistance and high reliabilities. With changing communication environments from peer to peer topology to bus topology, we should check each devices about not only mechanical operations but also electronic or software operations. In this paper, we suggest reliability test environment for CAN based system, which is divided two parts, data correctness and timely delivery.

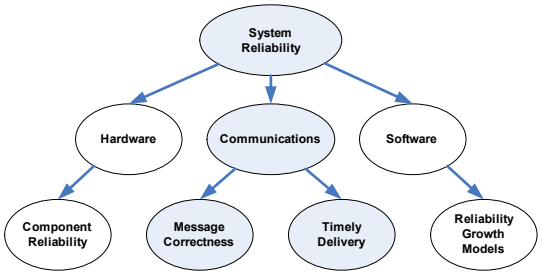
Key Words : Embedded system, Controller Area Network, Reliability, In-vehicle communication network

1. 서 론

CAN은 Controller Area Network의 약어로서 차량 내에서의 정보전달을 위해 개발되었다. 차량에서 사용되는 장치의 수가 증가하고, 이들 간의 정보교환이 필요해 지면서 효율적인 통신환경을 제공하기 위해 CAN이 등장하게 되었다. 차량의 운행에 필요한 많은 정보들이 CAN을 통해 전송되면서 전송되는 데이터에 오류가 발생할 경우에는 심각한 결과를 초래할 수도 있게 되었다. 따라서 장치에 대한 신뢰성 검증이 중요한 주제로 떠올랐다.[2]

신뢰성이란 주어진 시간 동안, 주어진 조건 하에서 시스템이 정상적으로 동작할 확률로 정의 된다. 시스템의 신뢰성이란 하드웨어에 대한 신뢰성, 소프트웨어에 대한 신뢰성 그리고 통신에 대한 신뢰성으로 구별할 수 있다(그림1)[2]. 우리가 관심을 가지는 부분은 CAN을 이용한 통신에 대한 신뢰성 테스트이며, 이를 위해서는 메시지의 정확성과 시간지연에 대한 테스트가 수행되어야 한다.

CAN은 네트워크의 설계단계에서 이미 메시지의 ID, 전송시간, 데이터의 크기, 송수신node와 같은 요소들이 정의가 되며 이를 이용하여 메시지의 정확성과 시간지연에 대한 신뢰성 테스트를 진행할 수 있다[5]. 실제로 수학적인 방법을 통해서 시간지연에 대한 신뢰성을 검증하는 연구는 많은 부분이 진행되어왔다. 그러나 이론적인 연구에서는 CAN 메시지 자체에만 초점이 맞추어져 있어 네트워크를 구성하는 node들의 동작에 의해서 시간지연이 지켜지지 않는 경우에 대해서는 간과하고 있다.[3]



(그림 1) 시스템 신뢰성: Top-down view

※ 본 연구는 국가교통핵심기술개발사업에서 지원된 "신에너지 저상굴절차량 개발"과제의 일환으로 수행되었으며, 지원에 감사드립니다.
† 정 회 원 : 델파이코리아
†† 정 회 원 : 아주대학교 전자공학부 교수
††† 정 회 원 : 아주대학교 정보통신전문대학원 교수
논문접수 : 2006년 12월 13일, 심사완료 : 2007년 2월 1일

본 논문에서는 이론적인 계산이 아니라 실제적인 CAN 네트워크에 대한 검증을 통해서 시간지연뿐만 아니라 메시지의 정확성에 대한 신뢰성을 검증하는 방법을 소개하고자 한다.

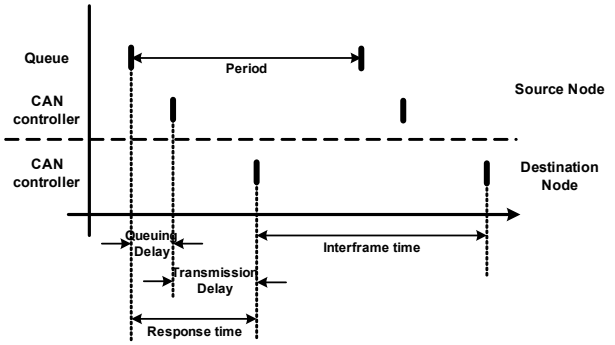
2. CAN (Controller Area Network)

CAN은 처음에는 차량용으로 사용할 것을 목적으로 설계되었지만 여러 가지 장점들 때문에 공장자동화나 항공기등과 같은 분야로 그 사용영역이 확대되어 가고 있다. CAN은 2.0A와 2.0B의 두 가지 구조를 가지고 있으며 이들의 차이는 단지 CAN 메시지의 ID 필드의 길이뿐이다. Media access control은 CSMA/CR(Carrier Sense Multiple Access/Collision Resolution)을 사용하며, CAN controller가 하드웨어적으로 오류를 검증하고 처리하는 구조로 되어있다[5].

3. 선행연구

CAN에서의 응답시간분석을 통한 지연시간의 신뢰성 검증에 대한 연구는 꾸준히 계속되고 있다[1,4,6,7,8]. [4]에서는 CAN에서의 응답시간분석을 위해 (그림 2)과 같은 model을 제시하고 있다.

응답시간(R_m)은 메시지가 생성되어 컨트롤러의 레지스터에 입력되는데 걸리는 시간인 큐잉지연시간(C_m)과 버스를 통해 전송되는 걸리는 시간인 전송지연시간(t_m)의 합으로 표현이 가능하다. 여기서 고려해야 할 사항은 메시지의 응답시간에 대한 Deadline(D_m)이다. 어떤 메시지가 주기적으로 발생한다고 가정하자. 만약에 응답시간이 어떠한 이유에 의해서 길어지게 되어 다음 큐잉시간까지 전송이 이루어지지 않을 경우 새로운 메시지가 컨트롤러의 레지스터에 덮어쓰여지게 되며 결국 그 메시지는 전송되지 못하고 버려지게 된다. 응답시간은 $R_m = t_m + C_m$ (1)로 정의 할 수 있다. 그러나 (1)은 전송 중에 발생하는 오류에 의해 재전송이 일어날 경우에 대한 부분이 빠져있다. 따라서 (1)식에 통신이 정상화되기까지 걸리는 recovery overhead를 반영하기 위해서 error function인 E(t)를 추가한다[6]. 이 후의 이론적인 연구들은 error function을 조금씩 개량하여 응답시간을 계산하는 것으로 EMI(Electro Magnetic Interference)에 의해 발생하는 오류들을 주기적으로 발생한다고 가정한 것[5], 오류발생 빈도가 특정 분포에 따르는 것으로 간주한 것[7], EMI의 발생 확률과 이로 인해 메시지에 오류가 발생할 확률을 정하여 계산한 것[8]이 있다.



(그림 2) 응답시간 분석을 위한 model

이론적인 연구에서는 CAN 메시지 자체에만 초점이 맞추어져 있어 네트워크를 구성하는 node들의 동작에 의해서 시간지연이 지켜지지 않는 경우에 대해서는 간과하고 있다. 발생한 오류들을 처리하기 위해서 CAN node들이 취하게 되는 동작이 메시지의 전송에 영향을 미치는 부분에 대해서는 생략하고 있는 것이다. 따라서 정확한 검증을 위해서는 실험적인 방법을 사용해야 한다.

4. CAN Fault Classes

CAN은 OSI 7 Layer에서 물리계층인 1계층과 데이터 링크 계층인 2계층에 해당한다. 3계층 이상의 상위계층은 Official standard로서 CAN기반의 네트워크에 대한 운영체제로서의 역할을 수행한다. Official standard에서는 메시지의 전송특성과 데이터의 크기등과 같은 네트워크의 특성들을 정의하고 있다. 즉, 2계층 이하의 경우에는 CAN 컨트롤러와 CAN 트렌시버를 이용하여 하드웨어적으로 구현되며, 3계층 이상의 경우에는 소프트웨어적으로 구현되는 것이다. 따라서 CAN에서 발생할 수 있는 오류는 크게 하드웨어에 의한 것과 소프트웨어에 의한 것으로 나눌 수 있다.

4.1 CAN 네트워크에서의 하드웨어에 의한 통신 오류

하드웨어에 의한 통신 오류의 경우 CAN 컨트롤러나 트렌시버 또는 CAN 버스의 오류에 의해 CAN frame에 오류가 발생하는 경우이다. 이러한 경우에는 CAN frame에서 오류가 발생하기 때문에 CAN 컨트롤러의 오류정정기능에 의해서 상위 계층으로 전달되지 않고 CAN 컨트롤러가 처리하게 된다. 따라서 소프트웨어적인 방법을 사용하여서는 이러한 오류를 찾아낼 수 없다. 하드웨어에 의한 통신 오류의 종류를 <표 1>에 나타내었다.

〈표 1〉 CAN fault classes(하드웨어)

세부사항	설명
Stuck-at fault	Node가 손상되어 constant bit value를 전송하는 것으로 다음과 같은 두 가지 종류로 나누어 진다. ● Stuck-at-recessive fault Recessive bit만을 bus로 전송하는 경우로 네트워크에는 피해를 입히지 않는다. ● Stuck-at-dominant fault Dominant bit만을 bus로 전송하는 경우로 네트워크에 치명적인 영향을 미치게 된다.
Shorted medium fault	Bus가 전기적으로 battery나 ground에 연결된 경우이다.
Medium partition fault	Bus가 끊어져서 서로 연결되지 않은 sub-network로 분리되는 경우로 정상적인 통신이 불가능해지는 경우이다.
Bit-flipping fault	Node자체의 문제나 connection medium의 문제로 node가 통제불능상태가 되어 에러가 있는 메시지를 전송하거나 정의되지 않은 value를 전송하는 경우

4.2 CAN 네트워크에서의 소프트웨어에 의한 통신 오류

소프트웨어에 의한 통신 오류의 경우는 CAN frame에는 문제가 없으나 소프트웨어의 문제로 ID, IDE, DLC, RTR등과 같은 메시지의 내용이 잘못되거나, 정해진 전송시간에서 벗어나거나, 같은 메시지를 무한이 반복하여 보내는 경우이다. Frame에는 문제가 없기 때문에 하드웨어적으로는 오류를 찾아낼 수 없으며, 메시지는 상위 계층으로 전달되기 때문에 소프트웨어적인 방법에 의해서 오류를 찾아야만 한다. 이러한 오류의 종류를 <표 2>에 나타내었다.

〈표 2〉 CAN fault classes(소프트웨어)

세부사항	설명
잘못된 value	ID, IDE, DLC, RTR이 잘못된 값을 가지는 경우
전송시간 오류	정해진 전송시간 내에 전송되지 않거나 너무 자주 전송되는 경우
Babbling-idiot fault	한 node가 time domain상에서 오류가 있는 메시지를 전송할 때 발생하는 fault로, 이로 인해 실제로 필요한 자원보다 더 많은 자원을 소비하여 다른 node들이 자원을 사용하지 못하게 하는 fault. 예를 들어 소프트웨어적인 오류로 인해 메시지를 무한 반복하여 전송하게 되는 경우

4.3 오류가 네트워크에 미치는 영향

하드웨어나 소프트웨어에 의해서 발생한 오류가 네트워크에 미치는 영향은 <표 3>과 같이 6가지로 분류가 가능하다.

〈표 3〉 CAN fault가 네트워크에 미치는 영향

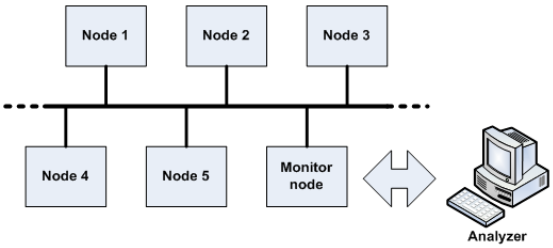
종류	설명
No effect	메시지의 전송에 아무런 오류가 없이 정확히 전송되고 수신된 경우
Performance Degradation	메시지에 오류가 발생하여 예상되는 전송시간보다 더 늦게 도착하는 경우(예 retransmission이 발생하는 경우)
Missing Frame	수신해야 할 메시지를 놓치는 경우(예 ID의 오류)
Incorrect Answer	잘못된 data를 수신한 경우로 CAN protocol이 오류의 존재를 체크하지 못하는 경우(예 전송할 메시지의 CRC계산 이전에 data에 오류가 발생할 경우)
Blocked node	오류가 발생한 node가 bus상으로 어떠한 메시지도 내보내지 않는 경우로 다른 node들은 정상적으로 메시지를 교환하는 경우
Blocked bus	bus가 block되는 경우로 어떠한 node도 통신을 할 수 없는 경우

5. 통신 신뢰성 테스트를 위한 환경

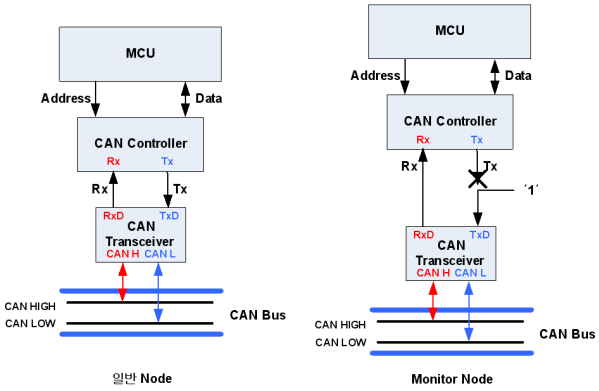
서두에서 설명한 바와 같이 논문의 목적은 CAN기반 시스템의 통신 신뢰성 검증이며, 이는 메시지의 정확성과 시간지연으로 나누어 진다. 이번 장에서는 통신 신뢰성 테스트를 위해 필요한 요소들에 대해서 살펴본다.

5.1 Monitoring node

Monitoring node는 CAN bus상에서 전송되는 메시지들을 수집하고 분석하는 기능을 수행한다(그림 3). Monitoring node는 CAN 네트워크의 모든 메시지를 수신할 수 있어야



(그림 3) 네트워크 구성도



(그림 4) 일반 node와 Monitoring node의 구성

하며, 수신한 메시지들에 대해서는 ACK를 하지 말아야 한다. Monitoring node는 다른 node들의 동작에 영향을 주어서는 안되기 때문에 다른 모든 node들에게는 존재하지 않는 것처럼 보여야 한다.

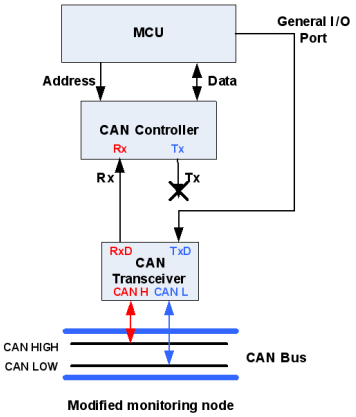
이러한 제약 조건을 만족시키기 위해서 monitoring node는 (그림 4)와 같은 방법으로 구성하여야 한다. 이와 같이 node를 구성하면 monitoring node의 acceptance filter를 설정하여 bus상의 모든 메시지를 수신할 수 있으며, 컨트롤러와 트렌시버 사이의 Tx연결이 제거되어 있기 때문에 monitoring node가 ACK를 전송할 수 없게 된다. 여기서 CAN 버스가 idle상태일 때는 recessive상태가 되어야 하기 때문에 트렌시버의 TxD단자에는 항상 '1'을 입력하여야 한다.

5.2 Message Information Table

CAN은 네트워크를 설계하는 단계에서 모든 node와 메시지에 대한 특성들이 정해지게 된다. Monitoring node는 이러한 정보를 가지고 자신이 수신한 메시지와 비교하여 오류의 존재 유무를 파악하고 어떤 node가 이러한 메시지를 전송하고 있는지 파악할 수 있다. 즉, 메시지의 ID가 확인되면 네트워크 설계단계에서 정의된 해당 메시지의 모든 특성들을 알 수 있으며 이를 monitoring node로부터 수신한 메시지와 비교하여 메시지의 정확성과 지연시간을 검사할 수 있다.

5.3 통신 오류의 생성

통신에서의 오류는 두 가지의 관점에서 만들어져야 한다. 각 node들의 소프트웨어적인 오류에 의하여 메시지의 내용에 오류가 발생하는 것과 컨트롤러나 트렌시버 또는 버스의 하드웨어적인 오류에 의해 나타나는 frame form에 대한 오



(그림 5) 수정된 monitoring node

류가 그것이다. 전자는 데이터 링크 계층 위에 일정한 확률로 메시지의 내용을 변경하는 오류 주입 계층을 추가하여 구현하는 것이 가능하다.

후자의 경우에는 버스상에서 전송되고 있는 frame 자체에 오류가 발생하도록 만들어주어야만 한다. 따라서 CAN 버스에 전기적인 오류를 만들어주어 구현하는 것이 가능하다. 이를 위해서 monitoring node를 (그림 5)와 같이 수정할 필요가 있다. 이는 평소에는 MCU(Micro Control Unit)의 general I/O port를 "high"로 하여 버스에 recessive bit를 전송하고, 오류를 만들어야 할 시기에는 port를 "low"로 하여 버스를 강제로 dominant state로 바꾸어 준다.

6. 실험 및 분석

실험에서는 SAE(Society of Automotive Engineers)[9]에서 제시한 Benchmark signal[10]을 이용하였다. Benchmark signal은 총 7개의 node와 53개의 signal로 구성되어있으며, 차량에서 "Battery", "Vehicle Controller", "Inverter/Motor Controller", "Instrument Panel Display", "Driver Input", "Brakes", "Transmission Control"을 담당하는 ECU들간의 signal들을 정의하고 이들의 전송특성과 데이터의 크기, 응답시간에 대한 Deadline등을 정의하고 있다. 이 signal들을 이용하여 정확한 실험을 하기 위해서는 Monitoring node를 포함하여 총 8개의 CAN node가 필요하지만 확보하고 있는 node의 수에 제한이 있어 5개의 node로 실험을 진행하였다. 이를 위해서 <표 4>와 같이 재배열되었으며, 전송주기가 짧은 메시지가 가장 우선순위가 높도록 메시지의 ID를 부여하였다. 실험은 메시지의 정확성에 대한 신뢰성 검증과 시간지연에 대한 신뢰성 검증을 따로 실시하였다.

〈표 4〉 Node별 송신 메시지와 ID

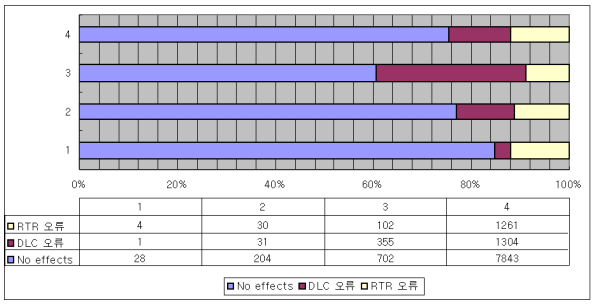
Node 번호	송신메시지(ID)	주기
1	2(16),14(17),17(18)	1000ms
2	1(13),6(14),13(15)	100ms
3	3(8),4(9),9(10),11(11),18(12)	50ms
4	5(1),7(7),8(2),10(3),12(4),15(5),16(6)	5,10,20ms

6.1 메시지의 정확성

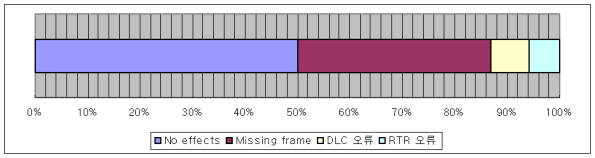
메시지의 정확성에 대한 신뢰성 검사를 위해서 모든 node들은 소프트웨어의 오류에 의한 메시지를 오류 주입 계층을 통해 ID, IDE, RTR, DLC에 각각 10%의 확률로 전송하였으며 이에 대해 monitoring node가 검사한 결과를 <표 5>에 나타내었다.

Missing Frame은 수신해야 할 메시지가 도착하지 아니한 경우로 ID 오류로 인해 발생한다. 전체 메시지 중 36.9%에서 이러한 오류가 발생하였다. Incorrect Answer는 CAN frame에는 오류가 없으나 그 내용이 잘못 전달되는 경우로 RTR, DLC, IDE에 오류가 발생한 경우에 해당한다. 전체 메시지 중 13%에서 이러한 오류가 발생하였다. Blocked node는 node에 오류가 발생하여 더 이상 메시지를 전달하지 못하는 경우이다. 이러한 경우는 실험 중에는 발생하지 않았으나 하드웨어적인 오류에 의해서 충분히 발생할 수 있는 현상이며, 응용프로그램의 오류에 의해서도 발생할 수 있는 현상으로 시스템에 치명적인 문제를 일으킬 수 있다.

〈표 5〉 Node별 송신 메시지와 오류발생 비율



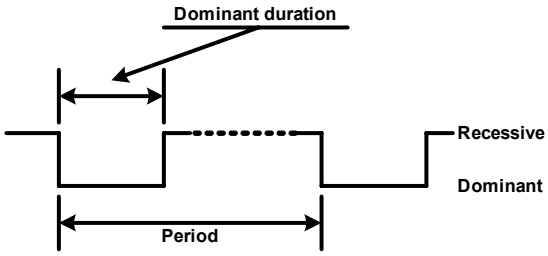
	1	2	3	4
RTR 오류	4	30	102	1261
DLC 오류	1	31	355	1304
No effects	28	204	702	7843



6.2 지연시간

지연시간에 대한 신뢰성 검증이란 각 메시지들이 정해진 응답시간에 대한 deadline을 넘기지 않고 전송이 되고 있는지 확인하는 것이다. 여기에는 여러 가지 원인이 있을 수 있다. 응용프로그램의 오류에 의해서 정확하게 전송이 되지 않는 경우도 있을 수 있으며, 소자의 결함으로 인해서 그러한 현상이 발생할 수도 있다. 하지만 차량이나 공장과 같은 CAN이 사용되는 환경을 고려해 본다면 이것들에 의한 오류보다는 EMI등과 같이 잡음을 발생시키는 요소들에 의해서 전송 중인 메시지에 오류가 발생하고 이를 다시 전송하는 과정에서 전송시간에 오류가 발생할 가능성이 대단히 높다고 할 수 있다. 응답시간은 큐잉지연시간과 전송지연시간의 합으로 표현된다. 이 중에서 외부 환경에 의해서 가장 영향을 받기 쉬운 것은 전송지연시간부분으로 (그림 5)에서 제시한 환경으로 조절이 가능하다. (그림 6)는 (그림 5)을 이용하여 생성할 수 있는 오류를 나타내고 있다.

즉, 일정한 주기로 임의의 시간 동안 bus의 상태를



(그림 6) 수정된 Monitoring node를 이용하여 강제로 bus에 오류 생성

dominant가 되도록 하는 것이다.

실험은 <표 4>와 같은 환경에서 응용프로그램의 오류는 없다는 가정하에서 50ms의 주기로 2.7ms, 5.6ms, 10.4ms의 dominant duration으로 설정하여 전송한다. 이에 대한 결과는 <표 6>, <표 7>, <표 8>에 나타내었다.

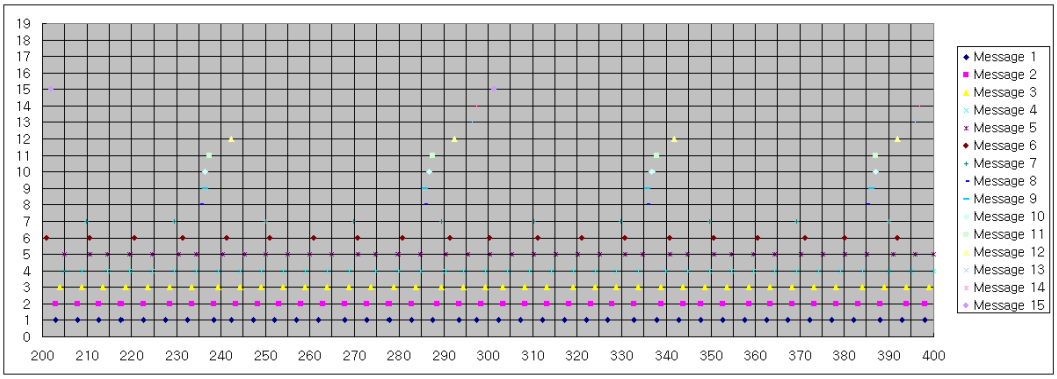
Dominant duration은 최소 주기인 5ms을 기준으로 2.7ms와 10.4ms으로 실험하였다. 각 표에서 가로축은 시간을 나타내며 ms단위로 표시되었으며, 세로축은 메시지의 ID로 <표 4>에 나타낸 것과 같이 정의 되어 있다. ID 16이상의 경우 주기가 1초이기 때문에 그래프상에는 나타나지 않았다. <표 6>의 경우 dominant duration이 짧기 때문에 메시지의 전송에는 아무런 영향을 주지 않음을 알 수 있다. 그러나 <표 7>와 <표 8>에서는 dominant duration이 길어지면서 그 영향이 나타남을 알 수 있다. 여기서 주목해야 하는 사항은 <표 7>과 <표 8>에서 dominant duration구간이 끝난 후에 일정 시간 동안 ID 1부터 7까지의 전송이 중단 된다는 점이다. 이러한 현상은 <표 8>에서 극명하게 드러난다. 그 이유는 CAN 컨트롤러의 오류정정기능 때문이다. ID 1에서 7은 하나의 node에서 전송하는 메시지들로 주기 또한 짧다. 따라서 error count가 빠르게 증가하면서 error passive mode로 동작하게 되고 메시지의 전송은 중단되게 된다. 이 node가 다시 메시지를 전송하기 위해서는 error active mode로 돌아와야만 하고 이 시간이 지난 후에 메시지 전송은 재개된다. 따라서 초기 네트워크의 설계 단계에서 메시지들을 적절히 분배하여 하나의 node가 순간적으로 많은 메시지를 전송하는 경우가 발생하지 않도록 해야 한다.

7. 결론 및 향후 연구과제

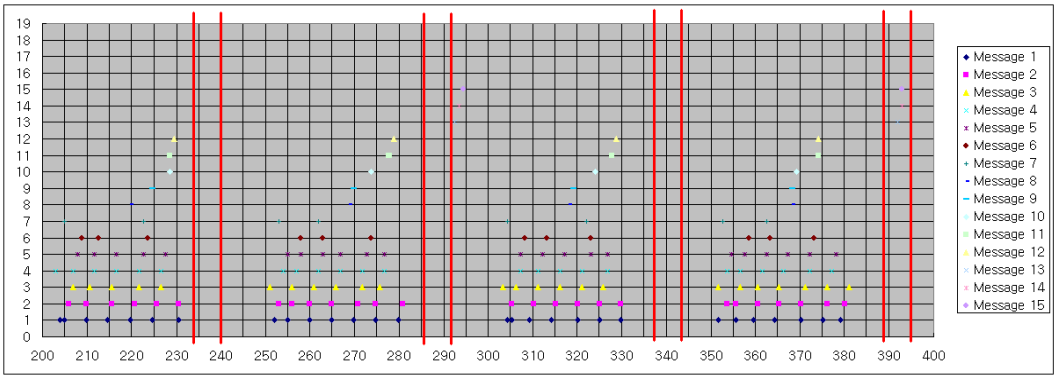
실험 결과에서 본 바와 같이 CAN 네트워크가 <표 4>와 같이 구성되어 있었다면 5.6ms정도의 길이로 EMI가 발생하여 통신에 장애가 발생하면 node 4는 block되고 7종류의 메시지가 전송시간에 대한 deadline을 지키지 못하는 결과를 나타내게 된다. 따라서 이는 잘못된 설계임을 알 수 있다.

시스템의 신뢰성을 검증하는 하는 것은 3 부분으로 나누

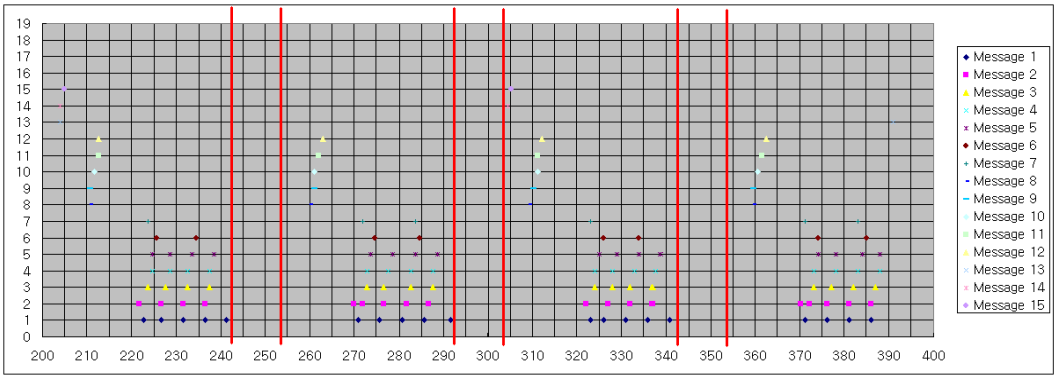
〈표 6〉 50ms주기, 2.7ms dominant duration



〈표 7〉 50ms주기, 5.6ms dominant duration



〈표 8〉 50ms주기, 10.4ms dominant duration



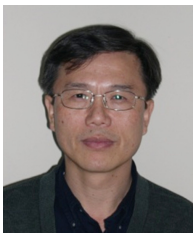
어진다. 그 중에서 본 논문에서 주제가 되는 부분은 통신에 대한 신뢰성 검증이다. 이는 다시 데이터의 내용에 대한 신뢰성과 전송시간에 대한 신뢰성에 대한 검증으로 나누어진다. 본 논문에서는 CAN은 네트워크의 설계단계에서 해당 네트워크에서 사용되는 메시지들의 ID와 전송시간, 전송방법 등의 특성이 모두 정의가 되어야만 하는 점을 이용하여 전송되는 메시지에 대한 신뢰성을 검증하였다. 메시지의 시간에 대한 신뢰성을 검증해 보려는 이론적인 시도는 많이 있었지만 하드웨어에 대한 고려가 없이 진행되어왔기 때문에 오류정정기능과 같은 동작에 의해 발생 할 수 있는 시간 지연은 무시되어왔다. 그러나 본 논문에서는 실제 네트워크를 구성하여 실험을 하였기 때문에 보다 정확한 실험을 할 수 있었으며, 이를 네트워크의 설계단계에서 사용하면 보다 신뢰성 있는 설계를 가능하게 할 것이다.

참 고 문 헌

[1] http://www.can-cia.org, 2005. Homepage of the organization CAN in Automation
[2] Thomas Nolte, Hans Hansson, and Christer Norstrom, “Probabilistic Worst-Case Response-Time Analysis for the Controller Area Network,” 9th IEEE Real-Time and Embedded Technology and Applications Symposium, 2003
[3] Julio Perez Acle, Matteo Sonza Reorda, Massimo Violante, “Early, Accurate Dependability Analysis of CAN-Based Networked Systems,” IEEE Design & Test, January-February 2006
[4] K. Tindell, A. Burns, A. J. Wellings, “Calculating Controller Area Network (CAN) Message Response Times,” Control Engineering Practice 1163-1169, 1995
[5] Wolfhard Lawrenz, “CAN System Engineering From Theory to Practical Application,” Springer, 1997
[6] Sasikumar Punnekkat, Hans Hansson and Christer Norstrom, “Response Time Analysis under Errors for CAN,” IEEE Real Time Technology and Applications Symposium, 2000
[7] Ian Broster, Alan Burns, Guillermo Rodriguez-Navas, “Probabilistic Analysis of CAN with Faults,” 23rd Real-Time Systems Symposium 2002
[8] Hans A. Hansson and Christer Norstrom, “Integrating Reliability and Timeing Analysis of CAN-Based Systems,” IEEE Transactions on Industrial Electronics, Vol.49, No.6, Dec 2002
[9] http://www.sae.org. Homepage of Society of Automotive Engineers
[10] SAE. Class C Application Requirement Considerations-SAE J2056/1. SAE Handbook, pages 263-271, Oct 1992

김 종 현

e-mail : toy3901@hotmail.com
2005년 아주대학교 정보통신대학 전자공학부(학사)
2007년 아주대학교 정보통신대학 전자공학과(석사)
2007년~현 재 델파이코리아
관심분야: 임베디드시스템, CAN 통신



정 기 현

e-mail : khchung@ajou.ac.kr
1984년 서강대학교 공과대학 전자공학과(학사)
1988년 미국일리노이대학 EECS(석사)
1990년 미국퍼듀대학교 EE(박사)
1991년~1992년 현대전자반도체연구소
1992년~현 재 아주대학교 전자공학부 교수
관심분야 : 인터넷보안, 임베디드 시스템

최 경 희

e-mail : khchoi@ajou.ac.kr
1984년 서울대학교 사범대학 수학교육과(학사)
1988년 프랑스 그랑데꼴 Enseeiht 정보과학과(석사)
1990년 프랑스 Paul Sabatier 정보과학과(박사)
1982년~현 재 아주대학교 정보통신전문대학원 교수
관심분야 : 운영체제 임베디드 시스템, 실시간 시스템