



AUTOSAR 4.1.1의 최신 기술과 이해


Part 5: AUTOSAR CAN

2014년 03월호 지면기사 / 글 | 이 현 철 <owen.lee@infobank.net> , Infobank

l5&f_size=13)  [프린트](#)

:/articleView.asp%3Fidx=1315&t=AUTOSAR%204.1.1의%20최신%20기술과%20이해) 

3D1315&text=AUTOSAR%204.1.1의%20최신%20기술과%20이해)  (https://story.kakao.com/s/share?

3D1315&text=AUTOSAR%204.1.1의%20최신%20기술과%20이해)  (https://story.kakao.com/s/share?
:AN&kakao_agent=sdk%2F1.37.2%20os%2Fjavascript%20lang%2Fko-KR%20device%2FWin32%20origin%2Fhttp%253A%252F%252Fwww.autoelectronics.co.kr)



Infobank 연구소 Smart Car Group AUTOSAR팀의 채승엽, 박대영, 이우승, 이현철 등이 AUTOSAR 최신 기술에 대한 아티클을 6회에 걸쳐 연재한다. 인포뱅크는 AUTOSAR 추세와 요구사항에 맞게 AUTOSAR Ethernet Solution, AUTOSAR 플랫폼 교육, SWC 개발 Tool과 같은 AUTOSAR 전반에 걸친 업무를 수행하고 있다. 제5회는 ‘AUTOSAR CAN’이다.

1. AUTOSAR의 실무 이해
2. AUTOSAR OS
3. AUTOSAR RTE
4. AUTOSAR MCAL
5. AUTOSAR CAN
6. AUTOSAR FlexRay

이번 회에서는 차량용 네트워크에서 사용되고 있는 CAN(Controller Area Network) 통신에 대해 알아보겠다. CAN 통신은 엔진 제어, 전동 파워 스티어링 등 차량의 많은 ECU에서 사용 중인 ISO 표준 시리얼 통신 규격(ISO 11898 등)이다. CAN 통신은 차량에 ECU 증가와 더불어 ECU 간 통신을 위해 전기 배선(Wire Harness)이 증가하게 돼 차량 중량 및 비용 문제가 발생하게 되면서 이를 감소시키기 위해 ECU 간 접속을 네트워크화하기 위해 개발됐다. 이러한 CAN 통신은 노이즈에 강한 차동 통신, 오류 검출 / 통지 / 복구 기능, 고장의 봉입 등으로 고신뢰성을 가지고 있으며, 노드(Node) 추가에 용이하다. 통신 속도 또한 실시간 제어가 가능한 1 Mbps 고속 통신을 제공한다. 하지만 속도의 경우 그림 1과 같이 버스 길이에 따라 전송속도가 달라진다. 그러므로 배선 길이에 대한 고려가 필요하며 일반적으로 CAN의 경우 40 M 이내일 경우 안정적인 속도를 보장한다. CAN 노드의 경우, 자신의 고유 Message ID를 갖고 있으며 각 노드는 버스가 Idle 상태 일 때 데이터 전송이 가능하다 (Multi Master). 만약 복수의 노드에서 동시에 버스에 데이터를 전송할 때 충돌이 발생할 수 있는데, 이 경우 Message ID를 통해 우선순위가 결정되며, 수신하는 노드에서는 Message ID를 확인해 수신 노드에 해당이 되면 저장(해당하지 않을 경우에는 무시함)하고 Ack 신호를 발생시킨다. CAN은 ID bit 길이에 따라 복수의 버전이 존재하며, 그림 2와 같이 표준 포맷의 경우 11 bit, 확장 포맷의 경우 29 bit로 이뤄져 있다. 이외 CAN에 대해 Frame / Error 처리 등에 대해서는 본 연재에서는 생략하고 AUTOSAR CAN에 대해 알아보도록 하겠다.

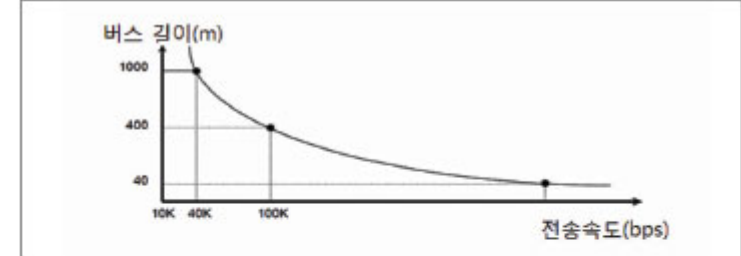


그림 1 | CAN 통신의 버스길이와 전송속도 관계 작성 Infobank

Version	2.0A	2.0B Passive	2.0B Active
ID의 bit길이	11bit	29bit	29bit
	29bit ID 수신시 error	송신은 11bit 29bit ID 수신시 무시	29bit ID 송수신가능

그림 2 | CAN Protocol Version 작성 Infobank

AUTOSAR에서의 CAN 통신

AUTOSAR를 적용해 CAN 설계에 대해 첫 연재에서 설명한 AUTOSAR Methodology를 기준으로 설명할 수 있다. AUTOSAR Methodology는 그림 3과 같이 시스템 설계 단계, ECU 설정 단계, BSW Generator 단계로 구성돼 있으며 CAN을 구성하는 데 있어 1단계인 시스템 설계 단계에서는 여러 개의 ECU가 어떻게 통신을 할 것인지를 설계한다. 2단계인 ECU 설정 단계에서는 1개의 ECU에 대한 BSW를 선택하고 상세 설계를 한다. 마지막인 3단계에서는 1개의 ECU에 대해 각각 BSW가 2단계의 ECU 설정 값들을 기반으로 소스 코드가 생성된다. 하지만 소프트웨어 컴포넌트(SWC) 소스 코드와 BSW 중에 Service SWC에 대한 integration Code 영역 부분은 개발자가 소스 코드를 작성해줘야 한다. AUTOSAR Methodology에 대해 자세한 사항은 첫 연재 자료인 “1. AUTOSAR의 실무 이해”를 참고하길 바란다.

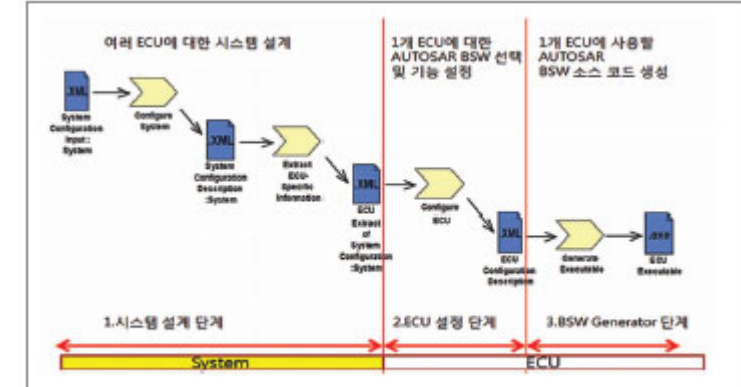


그림 3 | AUTOSAR Methodology 작성 Infobank


AUTOSAR에서 CAN Network 통신 절차에 대해 살펴보겠다. 그림 4에서 보는 것과 같이 CAN으로 데이터를 보낼 경우 SW-C에서 RTE, COM, PduR, CanIf를 거쳐 데이터를 전송하고, 이후 Can Driver에서 CAN Bus로 데이터를 전송하게 되고, 이렇게 전송된 데이터는 CAN Bus에 연결돼 있는 모든 ECU의 Can Driver에서 데이터를 수신하게 되며, 위에서 언급한 것


(http://ssl.logger.co.kr/tracker_ad.tsp?u=37061&mode=C&adCode=57236)

(http://ssl.logger.co.kr/tracker_ad.tsp?u=37061&mode=C&adCode=77860)

과월호 e-Book 보기

(/ebook/list.asp)





(/ebook/list.asp)(/ebook/list.asp)

News & Analysis

2륜차 리어램프에 최적! 4ch 리니어 LED 드라이버

(/article/articleView.asp?idx=3473)

다쏘시스템, 조메트리와 파트너십 체결...

(/article/articleView.asp?idx=3472)

포레스터 리서치, 산업용 IoT SW 플랫폼 리더로 마인드스피어 선정

(/article/articleView.asp?idx=3471)

[보도자료]ST, IIoT와 자동차 애플리케이션을 위한 안전한 셀룰러 연결 제공

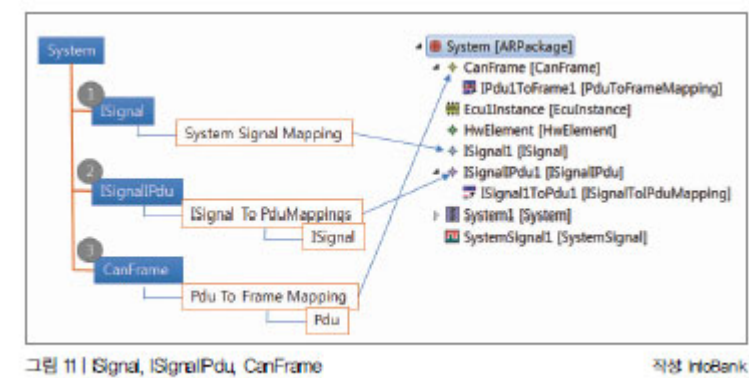
(/article/articleView.asp?idx=3470)

뤼츠 시스템 솔루션즈, 자동차 이더넷 테스트(ATE) 발표

(/article/articleView.asp?idx=3469)

(https://www.drivingthenation.com)

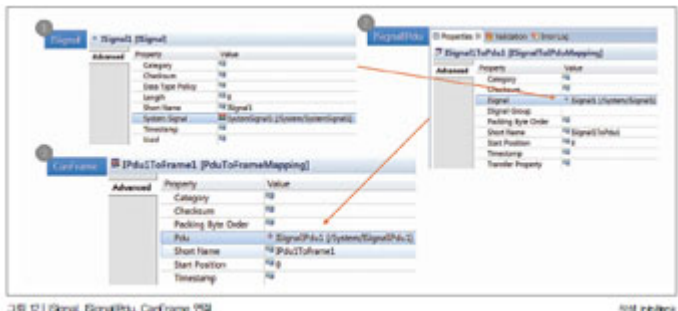
(http://smartn.co.kr/book/book_detail.asp?p_no=B00159)



AUTOSAR에서 CAN 시스템 설계

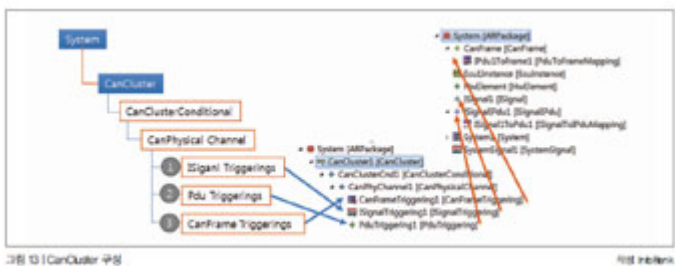
앞에서 AUTOSAR CAN에 대해 대략적으로 살펴보았다. 이제 시스템 설계 단계에서 CAN 통신을 하기 위해 Arxml을 설계하는 방법에 대해 살펴보았다.

그림 8은 Sender / Receiver interface를 사용해 ECU 간 통신으로 사용하기 위해 System Signal을 설정하고, 네트워크에 보내기 위해서 ISignal로 적용하고, ISignalPdu은 ISignal로 구성하고, Frame는 ISignalPdu으로 구성된다. 네트워크 설계를 하기 전 도식화된 컴포넌트에 대해 Artop을 통해 설계한 결과를 그림 9에서 확인할 수 있다. 설계된 내용을 간략하게 살펴보면, Application SWC인 Swc1에 InternalBehavior를 추가하고 함수(Runnable)인 F1을 생성했다. 이후 통신에 사용할 PPort를 추가해 Swc1에 대해 도식된 화면과 같이 구성을 완료한 것이다.



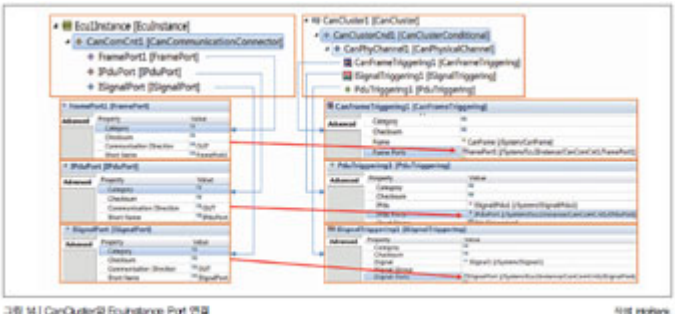
Composition SW-C를 생성해 Swc1에 대한 연결 관계를 설정한다.

위와 같이 설계된 CSW-C 파일을 가지고 시스템 설계를 진행해야 하는데, 우선 System Element를 추가한 후 그림 10과 같이 Root Sw Composition을 추가해 Composition SW-C와 Ecu Instance를 연결한다. 두 번째로 System Mapping을 추가해 Ecu와 SW-C를 Mapping하며 사용할 Ecu Instance와 H/W Element를 지정하게 된다. 만약 Ecu 내 통신을 하는 경우 이면, SwcTo EcuMapping 까지만 설정해 주면 되지만 현재 우리는 ECU 간 통신을 하기 위한 설계를 진행 중이니 System Signal을 추가해 줘야 한다. ECU 간 통신을 하기 위해서 System Mapping의 Data Mappings에서 Signal Mapping을 추가해 줘야 되는데, 이 때 Swc1에 맞는 interface 항목으로 Mapping을 해주면 된다. Sender Receiver interface를 사용하므로 “Sender Receiver To Signal Mapping”을 추가(그림 10에 4번 항목)한다. 이제 추가된 S/R To Signal Mapping에 System Signal을 지정해 주면 된다. SenderReceiver ToSignalMapping의 Properties에 System Signal을 설정해 주는 부분이 있다. (그림 10 붉은 네모 박스) 이 부분을 설정해 주면 Mapping이 된 것이다. 이제 그림 8에 System Signal까지 설계가 됐다.

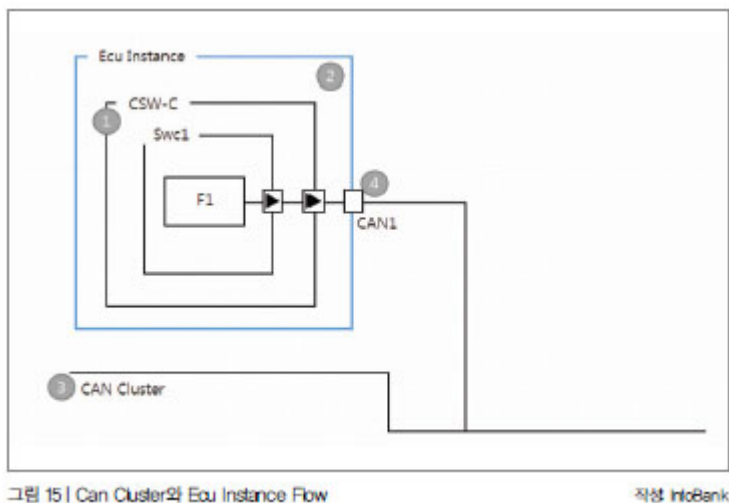


이제 Signal, PDU, Frame을 작성해 해 Mapping시켜줘야 하고, 설계 순서는 Signal → PDU → Frame 순으로 이뤄져야 한다. 각 항목에 대해 구성하게 되면 그림 11과 같이 설계가 이뤄지게 되는데 Element에서 ISignal을 추가해 위에서 작성했던 System Signal과 Mapping을 시켜줘야 한다.

그림 12에 1번은 그림 11 1번의 Properties 내용으로 위에서 설계한 System Signal에 ISignal에서 설정을 해주고 있다. 이렇게 ISignal을 설계한 후 2번과 같이 ISignalPdu를 추가해 1번에서 지정한 ISignal을 지정해 준다. 하나의 Pdu 안에는 여러 개의 Signal이 들어갈 수 있으며(그림 5 참조), 만약 여러 개의 ISignal을 추가해 줘야 되는 경우에는 그림 11에서 보는 것과 같이 ISignalToPdu Mapping을 Signal만큼 생성해 Mapping을 해줘야 한다. 하지만 현재 예제에서는 1개의 Signal로만 구성돼 있으므로 하나만 생성을 했다. 추가적으로 그림 12의 2번 항목에 보면 Start Position이 보일 것이다. 이 항목은 Pdu 안에서 Signal의 시작 위치를 지정하는 것으로, 만약 Signal이 두 개 있을 경우 Signal 1은 0부터 시작하고, Signal 2는 8부터 시작 하는 것으로 구성할 수 있다. 여기에서는 한 개의 Signal이므로 0이라 지정했다. 이제 Can Frame을 추가해 Pdu와 Mapping을 시켜줘야 한다(사용하는 통신에 따라 Lin Frame, FlexRay Frame을 추가하면 된다). Can Frame도 그림 12의 3번과 같이 2번에서 생성된 Pdu를 Pdu To Frame Mapping을 통해 설정하고 Start Position을 지정해 주어야 한다.



이제 차량 네트워크 통신을 하려면 Cluster와 EcuInstance에서 Connector를 추가해줘야 한다. Cluster는 Can의 속도, Type 등을 설정해 줄 수 있으며 모든 ECU는 Cluster를 경유해서 데이터를 주고받고, EcuInstance의 CanCommunicationConnector의 ISignal Port, IPduPort, FramePort 정보와 ISignal, ISignalPdu, CanFrame이 상호 간에 연결돼 있어야 한다. CanCommunication Connector는 통신의 송수신 방향을 결정할 수 있고 Connector 한 개당 Can Driver 1개라고 생각할 수 있다. Can Cluster에 신호를 보내기 위해 ISignal Triggerings, Pdu Triggerings, CanFrame Triggerings에 설계해야 하며, CanCluster에 Triggering를 추가하려면 CanClusterConditional을 추가해 줘야 되는데 CanCluster Conditional에서 속도를 지정한 후 Channel을 통해 상기 3가지의 Triggerings를 추가한다. ISignal Triggerings은 위에서 설계한 ISignal1을 연결하고, Pdu Triggerings은 ISignalPdu1, CanFrameTriggerings은 CanFrame을 연결한다. 자세한 것은 그림 13의 연관관계에서 볼 수 있다.



이제 CanCluster에 포트 정보를 입력해야 되는데, 이 부분은 EcuInstance에 있는 CanCommunicationConnector에서 생성을 한 후 설정해 줘야 한다. CanCommunicationConnector 항목에 있는 Ecu Comm Port Instance에 보면 ISignal Port, IPduPort, FramePort가 존재하는데, 각 항목에 대해 추가한 후 Communication Direction에서 해당 Port에 대해 송수신 값에 맞게 IN(수신)/OUT(송신)을 설정한다. 이렇게 설정한 Connector의 각 Port들은 그림 13에 있는 Triggering과 Mapping을 한다. 즉, ISignalTriggering1의 ISignal Ports 값에 EcuInstance의 CanCommunicationConnector에 추가된 ISignalPort가 Mapping이 되는 것이다. Pdu Port와 Frame Port 역시 ISignal과 동일하게 각 Port를 지정해야 한다. 그림 14의 각 Triggering에 대한 Property 설정 값을 보여주고 있다. 각 Property를 살펴보면 Port 값의 경로에서 보는 것과 같이 EcuInstance와 연결돼 있는 것을 볼 수 있다. 이제 마지막으로 EcuInstance CanCommunicationController를 추가해

주면 된다. CanCommunicationConnector는 통신에 대한 송수신 방향을 결정한다면 CanCommu nicationController는 차 량 네트워크의 Controller 특징(Time Segment)을 결정한다고 볼 수 있다.

위와 같은 절차로 설계가 되면 한 개의 ECU에 대한 Arxml 파일이 생성(ECU Extract of System Description Arxml) 되고 1 단계가 완료된다. 이후 2단계에서는 1단계에서 생성한 Arxml 파일을 가지고 Import해 BSW을 선정해서 Configuration하 게 된다. 선택하는 BSW에 따라 Configuration이 다르기 때문에, 본 연제에서는 2단계 Configuration은 제외했다. 이렇게 2단계에서 생성되는 파일을 ECU Configuration Description Arxml이라 한다. 3단계에서는 앞에서 작업한 Arxml 파일을 가지고 코드를 생성하고 Service SW-C와 SW-C에 대해 코딩 작업을 진행하면 된다.

시스템 설계 예제를 진행해 보았듯이 한 개 Signal에 대해서만 언급을 했다. 하지만 ECU에서 CAN 통신을 사용하는 Signal이 한 개만 존재하는 것이 아니다. 앞에서 보았듯이 각 Signal, PDU, Frame과 ECU Instance, Cluster, interface 등 각 항목에 대해 서로 연관 관계를 가지고 있어 CAN 통신을 많이 하는 ECU의 경우 서로 Mapping을 하는데 있어 주의가 필 요하다.

정리해보면, 시스템 설계 단계는 Software Component Template과 System Template으로 구분이 되며, 그림 15에서 1번 에 해당되는 Software Component Template은 Runnable, Port, CSw-C가 구성된다. System Template(네트워크 설계)은 그림 15에서 2, 3, 4에 해당되는 사항으로 2번에서는 ISignal, ISignalIPdu, CanFrame(Can 통신)에 대해서 설계를 하고, 3 번에서는 Can Cluster, FlexRay Cluster, Ethernet Cluster 등 어떤 통신을 사용할 것인지 설정해야 한다. 2번과 3번 항목이 설정되면 4번 항목인 EcuInstance에서 Can Communication Connector의 Port 설정(IN/OUT)을 해 Mapping 해 설계하면 된다.

마치며

이번 호에는 AUTOSAR에서 CAN 통신을 하기 위해 System Template에서 설계하는 방법에 대해 알아봤다. 한 개의 Signal에 대해 설계를 해 단순하게 보이지만 신호가 많으면 설계와 Mapping에 혼돈이 있으므로 작업하는데 주의해야 한 다. 만약 기존의 CAN DB를 Import 할 경우에는 System Template에 대한 정보는 추가되지만, 이럴 경우에는 Software Component Template에 대한 설계가 필요하니 시스템 설계에 대한 연관 관계를 잘 알아두길 바란다.

다음에는 AUTOSAR FlexRay에 대해 다룰 것이다. FlexRay는 기존 CAN 통신 프로토콜의 Event Driven 방식의 처리에서 Timing Trigger 방식의 처리로 마이크로세컨드 시간 단위로 여러 ECU와 동기화 해 실시간을 보장한다. 이것을 X-By-Wire 라고 한다. AE

<저작권자(c)스마트앤컴퍼니. 무단전재-재배포금지>

100자평 쓰기

로그인

로그인후 입력하세요

등록

Advertising / Media Partnership / Sponsoring

(/member/inquiry.asp?sel_type=ad)

회사소개 (http://www.smartn.co.kr) 개인정보취급방침 (/member/protect.asp) 이메일주소 무단수집 거부 (/member/noemailcollect.asp)
온라인 문의 (/member/inquiry.asp) 정기구독 신청 (http://www.smartn.co.kr/book/book_detail.asp?p_no=B00033)
정기구독 주소변경 (/member/subs_edit.asp)

스마트앤컴퍼니(주) 대표이사 : 박성규 사업자등록번호 : 108-81-64739 통신판매업신고 : 2019-서울구로-2138호

서울특별시 구로구 디지털로34길 43, 607호(구로동, 코오롱싸이언스벨리1차) P: (Phone) 02-841-0017 F: (Fax) 02-841-0584 ✉ webmaster@smartn.co.kr

© Smart & Company