Q&A

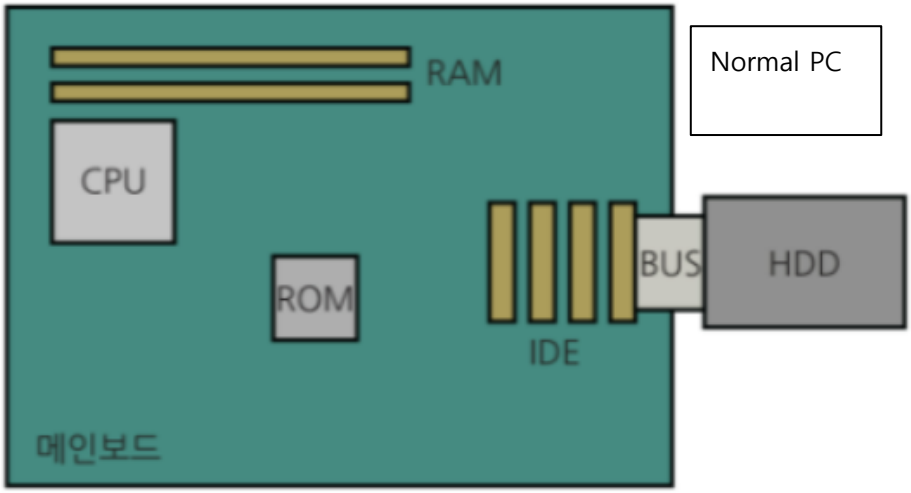
1. Difference between CPU and MCU

1.1. CPU = Central Processing Unit

1.1.1. Role : Arithmetic, logic, controlling, I/O(input/output) ***by instructions***

1.1.2. Component : ***ALU***(Arithmetic Logic Unit; Performs arithmetic and logic operations) + ***Processor registers***(Supply operands to the ALU and Store results of ALU operations) + ***Control Unit***(Fetching-Decoding-Execution Cycle)

CPU must be connected with Main memory(RAM) and HDD/SSD*

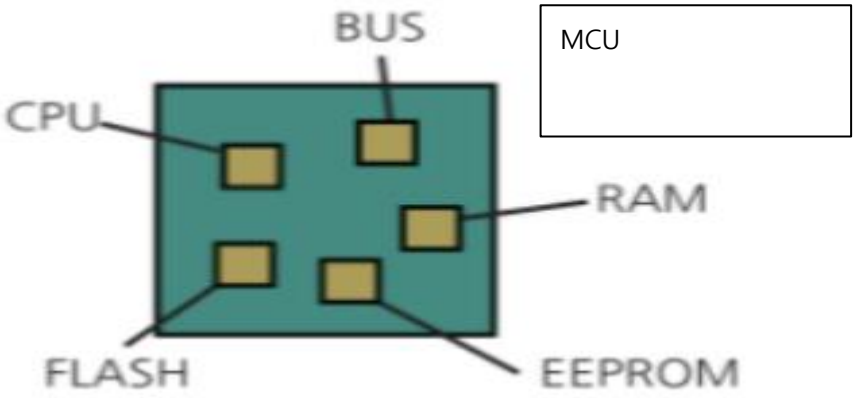


1.2. MCU = Micro Controll Unit = MICOM(MICro COMputer) = one-chip microcomputer

1.2.1. Role : Used in not many functional electronic device(e.g. Vehicle ECU) to do ***specific functions*** coded by machine language

1.2.2. Component : Compaction of ***real computer***(BUS, CPU, memory)

MCU is already connected with main memory(RAM) and HDD/SSD*



2. Basic concept of Firmware

2.1. Definition

-Firmware is a ***computer software*** that provides the ***low-level control*** for a device's specific hardware

-Firmware is a kind of ***OS***

-Firmware is stored in ROM(Read Only Memory)/PROM(Programmable Read Only Memory)

2.2. How to use?

-Injecting ***binary file***(language : C/ASM; Compiler : Embedded cross compiler) into EEPROM(Electrically Erasable Programmable Read Only Memory)

-In these days, Firmware loading is usually done in ***device driver*** because loading firmware on the host system is ***more flexible and cheap*** so that additional HW for firmware is not necessary.

Cf. 임베디드 시스템에서 bin이나 hex파일은 무엇인가?

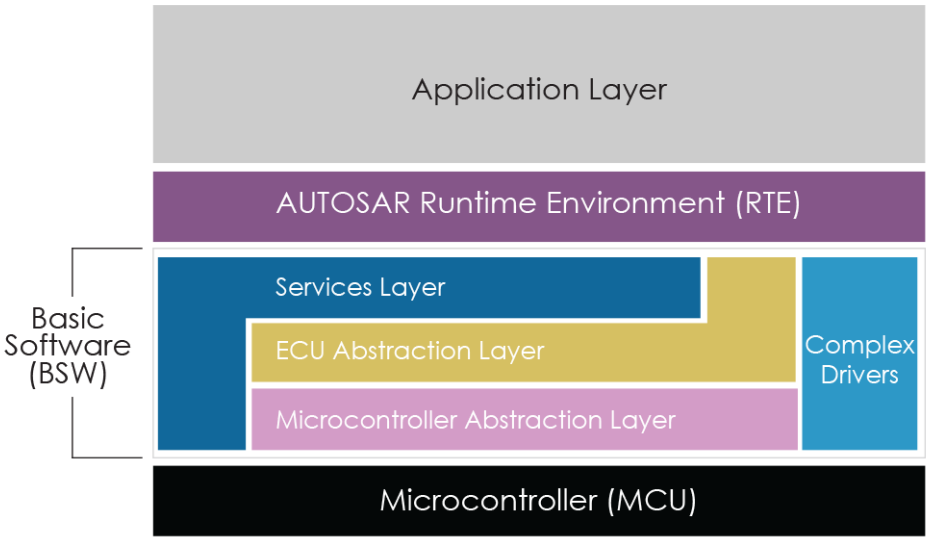
HW가 이해할 수 있는 언어는 기계어 뿐입니다. 이렇게 기계어로 이루어진 파일은 보통 ASCII 코드로 변환하여 .bin이나 .hex(.bin파일을 hex로 바꾼 후 ASCII 코드로 변환한 파일)의 확장자를 지니게 됩니다. 이렇게 ASCII 코드로 변환해야 사람이 볼 수 있고, 추가적인 정보도 보낼 수 있기 때문이다. 또한 hex 파일에는 프로그램 시작 주소등의 정보 등이 담겨 있다. bin 파일은 00번지 부터 시작하는 것으로 가정해서 만들어서, 만약 00번지에서 시작하지 않는다면, 00부터 시작 주소 까지는 0x00이나 0xff 등으로 채워지게 된다.

만약 우리가 원하는 어떤 Specific function으로 동작하는 코드를 만들어 hw에 올리고 싶다면 High-level languge(C/C++)를 사용하여 코딩을 한 후 적절한 컴파일러(Keil MDK,

SW4STM32 등)을 이용하여 빌드를 하면 bin/hex 파일이 생성이 되는데 임베디드 시스템의 경우 이렇게 빌드된 바이너리 파일을 MCU에 다운로드 하면 된다.

2.3. Is Application of AUTOSAR Firmware?

-No. Before AUTOSAR, application should include application and firmware so that application must be changed if company or OEM might be changed. After AUTOSAR, this problem has vanished.



AUTOSAR Architecture

Cf. Reson of AUTOSAR`s appearance

As software of ECUs in vehicle getting various and complicated, there are growing number of problems. Before using AUTOSAR, Application and Firmware are not developed separately so that ***SW itself had to be changed to provide it to other companies***. And it costed a lot.

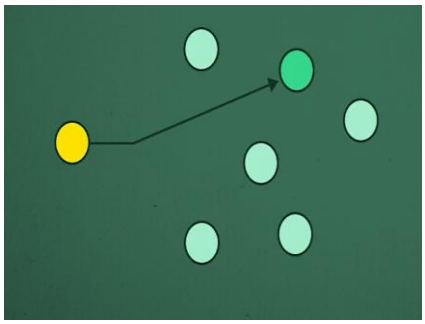
At first, it was not plausible to use AUTOSAR because they must change their previous processes at all to use it. But AUTOSAR was aiming for the share of benefit to everyone from small OEM/Supplier to big OEM/Supplier. It was necessary to make a ***general platform which can be used at any time***. So, they made AUTOSAR and now many company use AUTOSAR because of its benefits like cost-reduction, time-reduction when developing vehicle SW.

3. Difference between Unicast, Multicast and Broadcast

(1) Unicast

-A frame is sent from ***one ECU to another ECU***

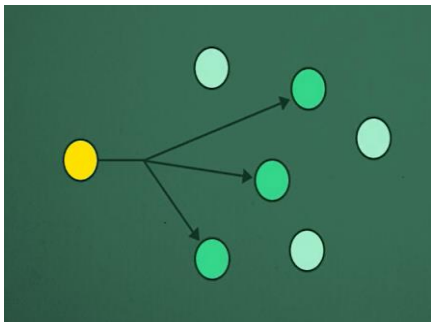
-One sender and One receiver



(2) Multicast

-A frame is sent from ***one ECU to several ECUs***

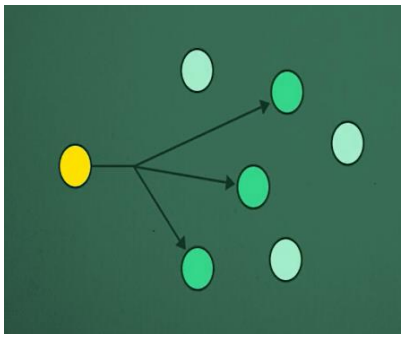
-One sender and Several receivers



(3) Broadcast

-A frame is sent from ***one ECU to all of the other ECUs***

-One sender and All of the other receivers



4. Hardware Abstraction

4.1. Definition

-Array of SWs play a role of ***bridge*** which makes ***OS to handle many HWs*** without any differences.

-Kernel which opens driver at system-booting

4.2. Property

-Prevents us from directly approaching to hw device

-Used as API

-***Not specific to hw device***. So, generally used

-Almost every OS has HA(Hardware Abstraction) in its kernel

-Positioned between SW and HW

5. Real-time requirements

5.1. Definition

Systems that work within ***strict time constraints*** and provide a ***worst case time estimate*** for critical situations. When there is an embedded component in a real time system, it is known as a real time embedded system.

5.2. Types of Real Time Embedded System

(1) Hard Real Time Embedded System

This type of system makes sure that ***all critical processes are completed within the given time frame***. This means that all the delays in the system are strictly time bound. Also, there is little to no secondary memory and data is stored in short term memory or read only memory. Hard real time systems are used in various areas such as missiles, airplanes etc.

(2) Soft Real Time Embedded System

These are much less constrictive than hard real time systems but the basic premise

is the same i.e critical processes need to be completed within the given time frame. However, this time frame can be a little ***flexible***. Soft real time systems are used in various areas such as multimedia, scientific projects etc.

6. Difference between Signal-based Communication and Event-based Communication

(1) Signal-based Communication

-Used in CAN, LIN, FlexRay, MOST

-Static communication of ECUs : Data is just sent over network whenever data values are modified ***without permissions*** of other nodes. It will eventually increase loads of unwanted data of nodes

-Using for SW which ***won`t be modified***

(2) Service-based communication

-Powered by Ethernet and SOME/IP, SOA(Service Oriented Architecture) models the entire system as service interfaces. ***New software can be easily added to the system*** without worrying about the compatibility with others. Ethernet provides backbone and UDP/TCP in transport layer and SOME/IP provides data serialization, remote call procedure for middleware

-Dymic communication of ECUs : Data is sent over network ***only if receivers want to receive.*** Server takes charge of checking notifications of receivers

-Using for automated driving, ADAS(Advanced Driver Assistance System), connected cars

(3) Event-based Communication

-Event-driven architecture(EDA) is a software architecture paradigm that considers ***the individual messages of a network of services to be events, rather than requests.***

-Events are considered as ***named state changes*** within the software or hardware

-EDA is split into event consumers and event producers. Event producers detect event and broadcast it as a message for consumers through event channels/streams. The consumers can either act on this event, or just be informed about it, depending on the functions of that consumer.

-EDA is constructed in such a way that events are moved around the system, and are stored in databases. A key benefit to storing data as events is that the context for each business decision is stored with the event.

7. Basics of RTE

7.1. Why do we use RTE?

RTE acts as a middleware between the AUTOSAR application layer and the lower layers. Basically, the RTE layer manages the ***inter- and intra-ECU communication*** between application layer components as well as between the BSW and the application layer.

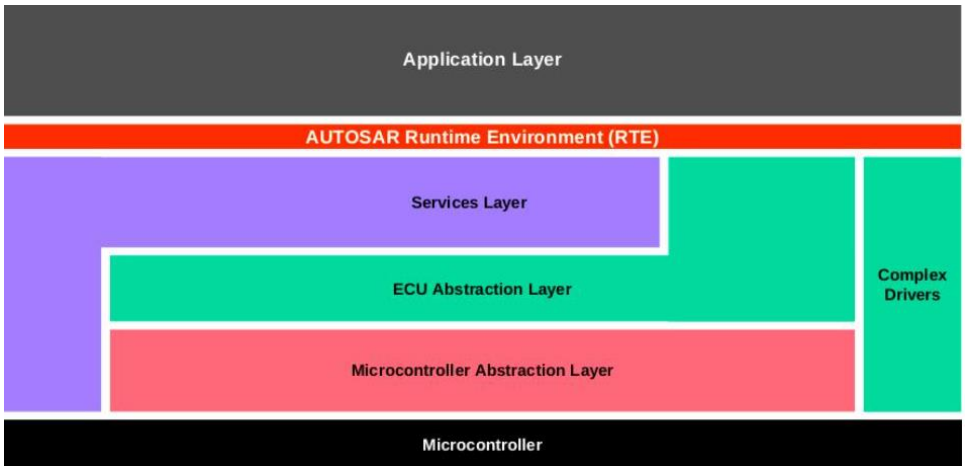
The interaction between ECU I and ECU II is an example of inter-ECU communication and takes place through and beneath the RTE and goes via the Basic Software Module. The BSW handles the functions like interactions with the memory along with communication services (if needed) for the inter-ECU communication.

The intra-ECU communication, that is the communication within the ECU between the software component B (SW-C B) and the software component C (SW-C C) , is entirely through the RTE. The software components that are mapped to a single ECU use the Intra-ECU method.

------ Communication Path

The implementation of the AUTOSAR software components is made independent of available communication mechanisms by the RTE by offering a uniform environment to these to AUTOSAR Software Components (SWC).

Application layer exchanges data with the underlying layers via the sender and receiver ports of the RTE. Whenever an AUTOSAR software component calls for the service objects, the RTE maps these requests to the actual service object symbols on the local ECU.



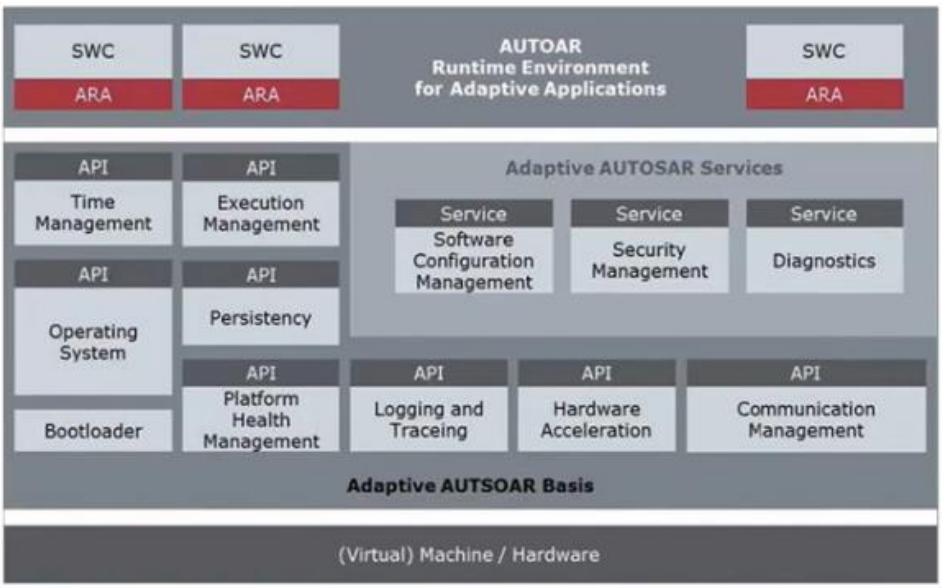
7.2. How we use RTE?

During design time of an AUTOSAR application, the ***VFB(Virtual Function Bus)*** is used to manage the communication between the software components. This virtual bus abstracts the applications. VFB is an abstract component that is represented usually by the Runtime Environment (RTE) at the runtime, and generated uniquely for each ECU in the AUTOSAR system.

8. Reasons why we are using Adaptive platform but not Classic platform

8.1. Adaptive platform의 개념

AUTOSAR 플랫폼은 소프트웨어 구동 전기 및 전자(SW driven E/E) 차량 아키텍처를 자동차 부문에 도입하여 확장해야 하는데 기본적으로 MCU가 어느 정도 성능 수준에서 끝나기 때문에 표준화를 더 강력한 MCU로 확장할 필요가 있었습니다. 헌데 원래 AUTOSAR 사양(현재의 Classic AUTOSAR)은 하나의 전용 ECU에 기능, 하드웨어 및 필수 소프트웨어가 함께 배치되어 구동되는 E/E 아키텍처용으로 설계되었는데 여기서는 더 강력한 MCU로 확장을 할 수 없다는 문제점이 발생하였고 이것을 해결하기 위해 Adaptive AUTOSAR는 기존의 Classic AUTOSAR 플랫폼을 확장하여 하나의 ***"차량 서버"***에서 서로 다른 기능을 위한 소프트웨어 기능을 결합하는 컴퓨팅 클러스터를 생성하는 소프트웨어에서 하드웨어를 분리하는 아키텍처로 설계하여 문제점을 해결하였습니다.



8.2. Adaptive platform의 이점

Adaptive AUTOSAR의 도입으로 우리는 모든 자동차 시스템에서 다음과 같은 많은 이점을 기대할 수 있습니다.

- 모든 제조업체를 위한 보다 효율적인 개발
- 개발 속도 향상
- 차량 하위 시스템 간의 인터페이스 개발 시간 단축
- 표준화를 통한 안전성 향상

제조업체에게 Adaptive Platform이 주는 이점

제조업체의 경우 adaptive autosar를 사용해서 차량 시스템을 더 쉽고 빠르게 개발 및 통합 가능해 졌습니다. 결과적으로 사용자는 여러 변형이나 사용자 지정 없이 조직 전체에서 하나의 공통 플랫폼을 사용할 수 있게 되어 AUTOSAR가 매우 개방적으로 확장될 수 있다는 것입니다.

소비자에게 Adaptive Platform이 주는 이점

안전하고 확실한 기반을 제공하여 표준화된 인터페이스를 사용함으로서 전체 플랫폼을 교체해야 한다는 시스템 전반의 버그와 오류가 줄어들 것입니다. 또한 다른 AUTOSAR 시스템과 더 잘 호환되어 최신 모델을 기존 인프라에 더 쉽게 통합할 수 있게 되었습니다.

시장에서 Adaptive AUTOSAR의 전망

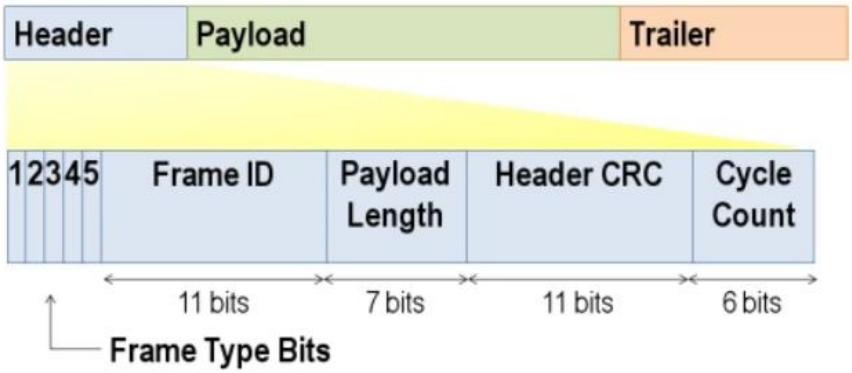
차량이 점점 더 스마트해짐에 따라 Adaptive AUTOSAR 시스템도 새로운 기술과 안전 규정을 지원하도록 업데이트되고 있으며 더 빠른 통신 프로토콜로 AUTOSAR 시스템을 업데이트함으로써 차량 제조업체는 비용을 줄이면서 성능을 높일 수 있게 되고 있습니다. 또한 고성능 컴퓨팅 플랫폼을 구현하면 성능이나 속도를 희생하지 않고도 여러 ECU를 사용할 수 있도록 Adaptive

AUTOSAR가 도움을 주므로 Adaptive AUTOSAR의 전망은 앞으로도 밝을 것으로 예측됨.

9. Data used in FlexRay

데이터의 ***오차가 허용되지 않고***, 데이터가 아주 ***빠르게*** 전송되어야 하는 상황에서 사용되는 데이터. 가령, 고성능 파워트레인, 안전 시스템(Drive-by-wire, 액티브 서스펜션, 적응 크루즈 컨트롤)에서 사용되는 데이터.

9-1. Header of FlexRay



헤더는 5 바이트 (40 비트) 길이이며 다음과 같은 필드를 포함합니다.

(1)상태 비트 - 5 비트 : State of data

(2)프레임 ID - 11 비트 : Slot carrying frame and used in prioritizing frame

(3)페이로드 길이 - 7 비트 : Length of words carred in frame

(4)헤더 CRC(Cyclic Redundancy Check) - 11 비트 : Detecting errors in carrying data

(5)주기 카운트 - 6 비트 : Increasing when communication period gets started

9-2. Payload of FlexRay

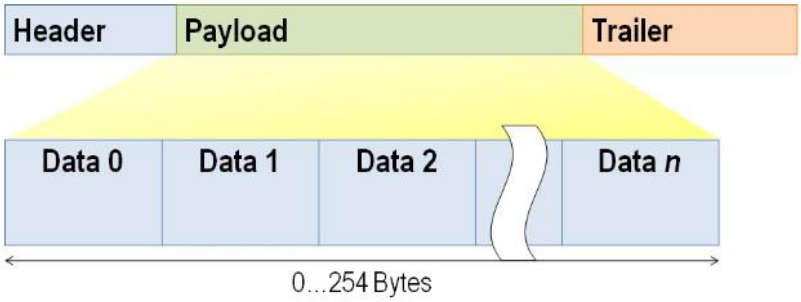
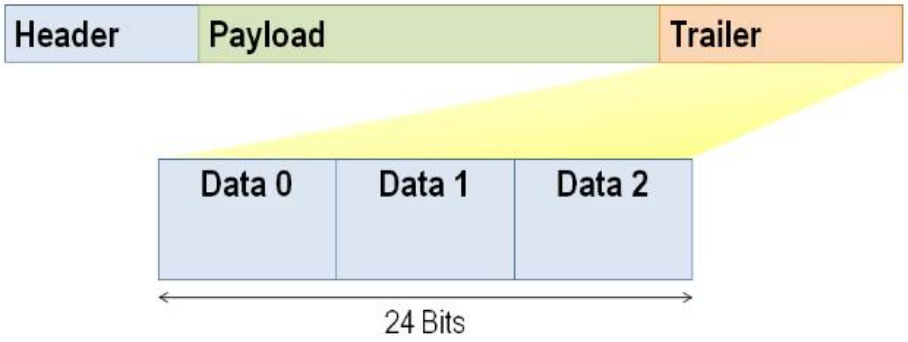


그림 9. FlexRay 프레임의 페이로드

페이로드에는 프레임이 전송하는 실제 데이터가 포함됩니다. FlexRay 페이로드 또는 데이터 프레임의 길이는 최고 127자 (254 바이트)이며 이는 CAN과 비교하여 30배 이상의 수치입니다.

9-3. Trailer of FlexRay

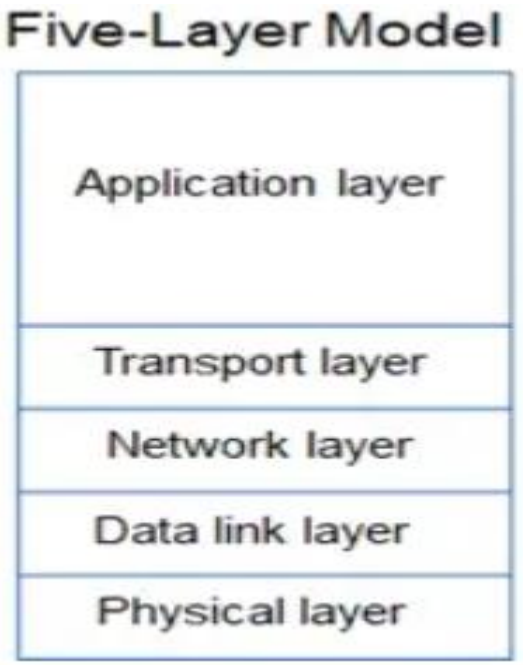


트레일러에는 에러 감지를 위한 세 개의 8-비트 CRC가 있습니다.

10. Difference between Network and Data Communication

(1)Data communications : ***Transmission*** of digital data between two or more computers.

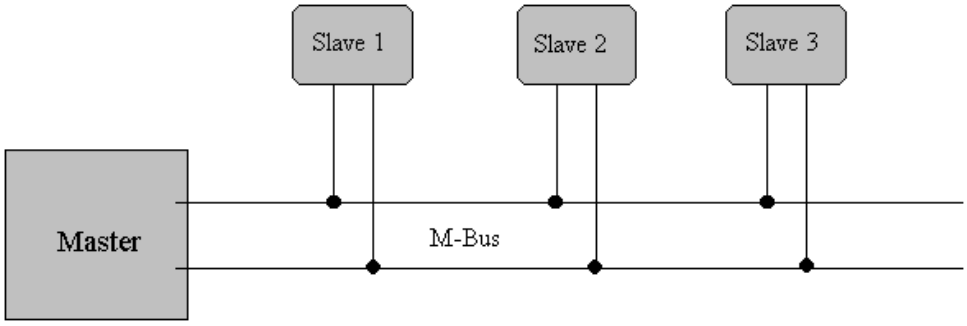
(2)Network : Telecommunications ***system*** that allows computers to exchange data. The physical connection between networked computing devices is established using either cable media or wireless media. The best-known computer network is the Internet.



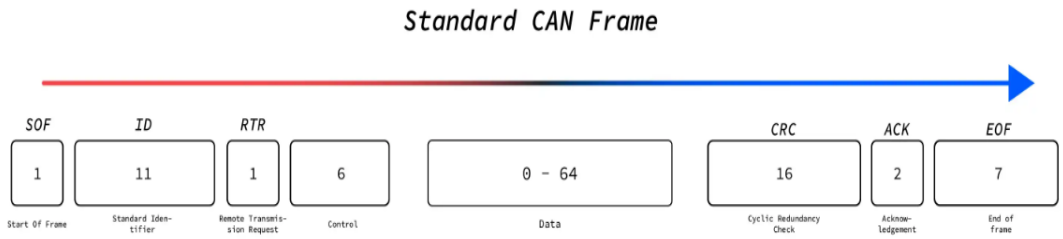
11. Difference between Master and Slave in BUS

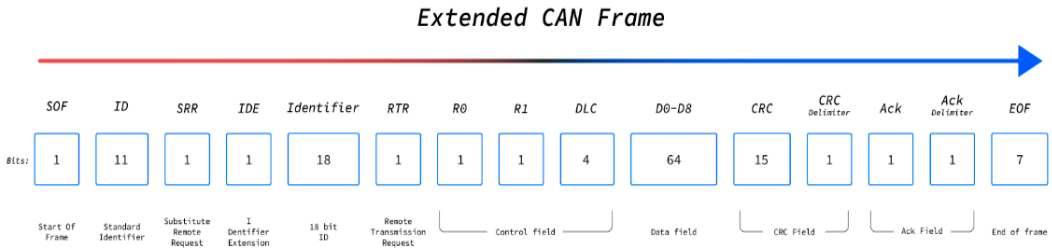
(1)Master : In asymmetric communication, Master is device/process which ***controls*** other device/process and bus pathways that contain the transmission signals and address .

(2)Slave : In asymmetric communication, Slave is device/process which ***responds to*** requests from master and understand how to communicate with other devices on a bus



12. CAN Frame





-SOF(Start Of Frame) : 1 bit; Representing the start of CAN Message

-ID : 11 bits; Organizing the priority of the CAN message. The smaller ID is, the higher priority it has.

-RTR(Remote Transmission Request) : 1bit;

Each other`s RTR	**Dominant(0)**	**Recessive(1)**
Dominant(0)	Dominant	Dominant
Recessive(0)	Dominant	Recessive

-IDE(Identifier Extension) : 1 bit; Dominant(0) when the standard CAN frame is sent – not extended CAN frame.

-DLC(Data Length Code) : 4 bits; Indicating how many bytes of data are in current message.

-Data : 64 bits; Transmitted data in the CAN message

-CRC(Cyclic Redundancy Check) : 16 bits; Checksum to detect errors and issues in the transmitted data

-ACK : 1 bit; In case the message is properly received, the receiving node will overwrite the recessive acknowledge bit with a dominant bit. Therefore if ACK bit was 0 when it arrived, then it becomes 1 when it is sent back.

-EOF(End Of Frame) : 7 bits; Indicating the end of the CAN message and detecting bit stuffing errors.

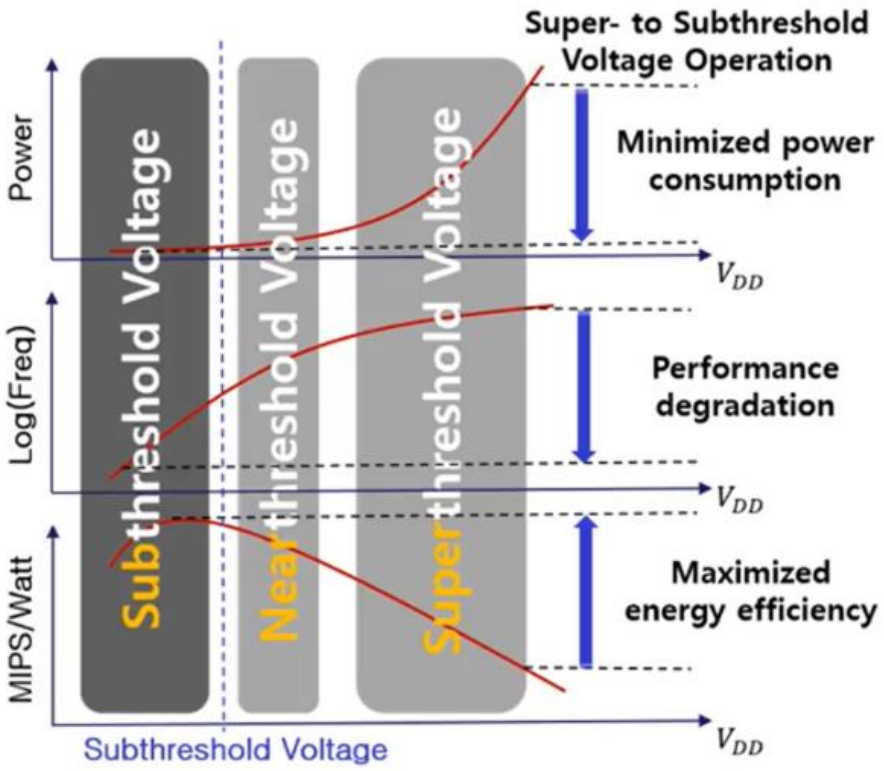
Cf. Why does standard CAN have 11bits of ID and extended CAN have 29 bits of ID?

Answer : CAN does not have limitations of numbers of ECUs wheter it`s standard CAN or extended CAN. Extended CAN is used in J1939 protocol - heave-duty vehicles and standard CAN is used in other vehicles. Their purposes of usage is different. That`s why.

13. Relationship of Performance and Power

-Power is ***heat*** and performance is depending on power. As scaling of semiconductor fabrication is getting decreased, the density of circuit is getting increased so that it is getting vulnerable to endure heat which finally leads to burning situation and also decrease performance.

I.e. Low power : It uses low voltage/heat so we can use battery for a long time.



14. Reasons why different vehicle networks are used

Unlike home network, vehicle network is very ***critical and important for our safety***. Home network doesn`t need to be sensitive for fault/error/malfunction. On the other hand, vehicle network needs to be very ***sensitive for fault/error/malfuction*** because it can lead us into facing life-threatening situation. So, It is important to ***categorize vehicle networks by their sensitivities.*** For example, LIN is the slowest but cheapest network so that it is used in the parts like mirrors and seats where network speed is not important. Ethernet is the fastest network so that it is used in the parts such as navigation. CAN-FD is an extension to the orignal CAN and it has better specs than CAN so it is now used in high performance vehicles like robotics, electric cars. CAN is the longest and the most sensitive network way. So, It is used medical device, aerospace device etc.

15. How to transferring data in CAN?

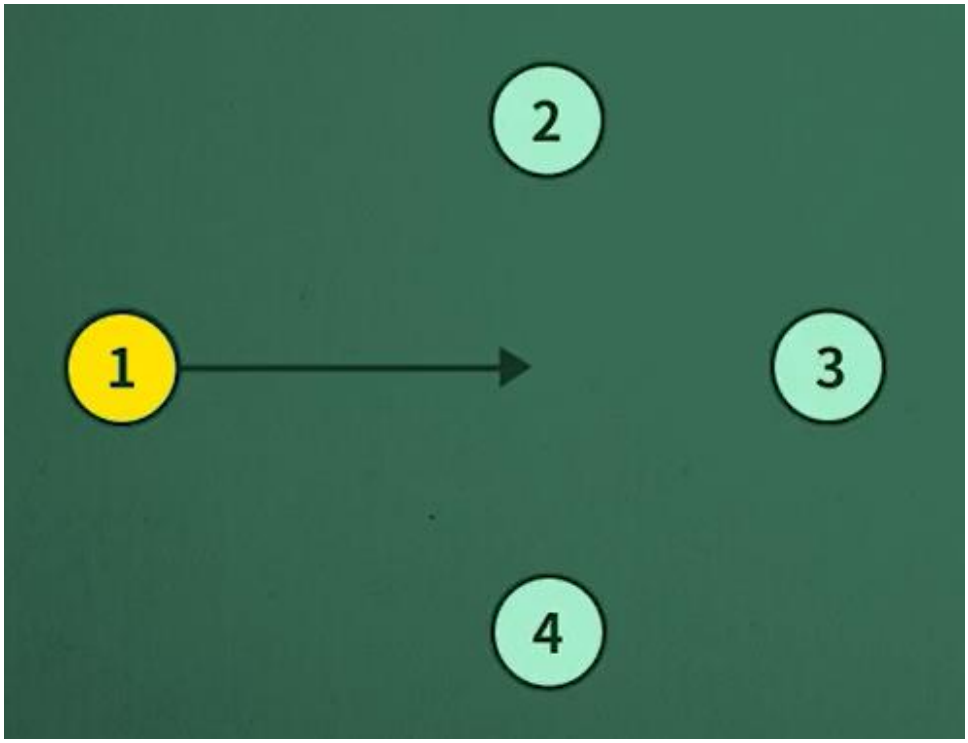
(1) CAN data frame

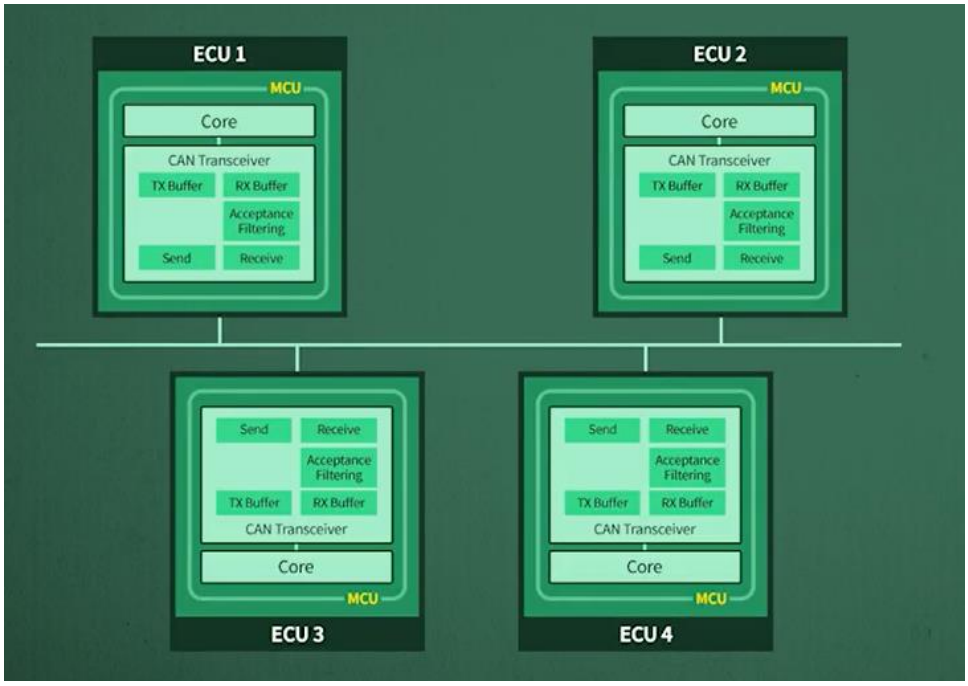
CAN ID doesn`t indicate the destination of CAN frame. But it only describes the ***meaning of data***.e.g. Engine rpm, Engine temperature, etc. Meaning of a message is marked by an identifier(ID). For example, A CAN network has 12 ECUs in 5 groups and 17 sensors are in ECU1.

There are 11bits of identifier.

(2)Process of CAN

Suppose that ECU1 sends one data frame of ID=0x1A to ECU2, ECU3 and ECU4.



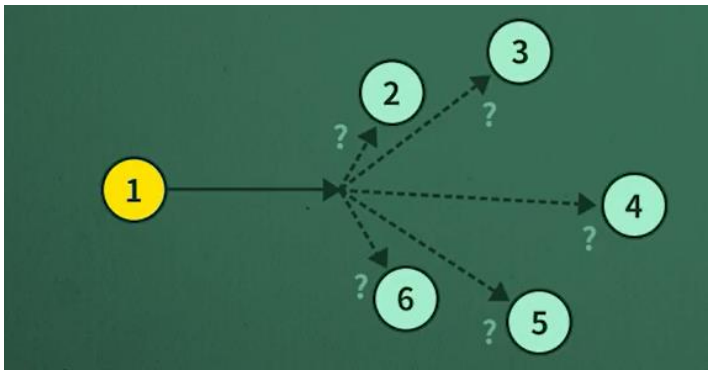


ID is in TX Buffer of ECU1 and it goes down to Send module and it will be transferred through copper wire. Receive module of ECU2, ECU3, ECU4 will get ID and they will decide wheter they will need to accept the ID by acceptance filtering and then ID goes to RX Buffer if they accept it. The data of packet will be extracted by them.

*Summary

CAN : ***Receiver-selective method***, Sender does NOT know its corresponding receiver and other ECUs except for sender will determine the extraction of information depending on ID and

acceptance filtering. And each ECU can decide the acceptance of the message carried on the bus by acceptance filtering



(3)Advantage of CAN

Increase configuration flexibility : Integration of ECU4 is very easy without modification of existing ECU1, ECU2, and ECU3.