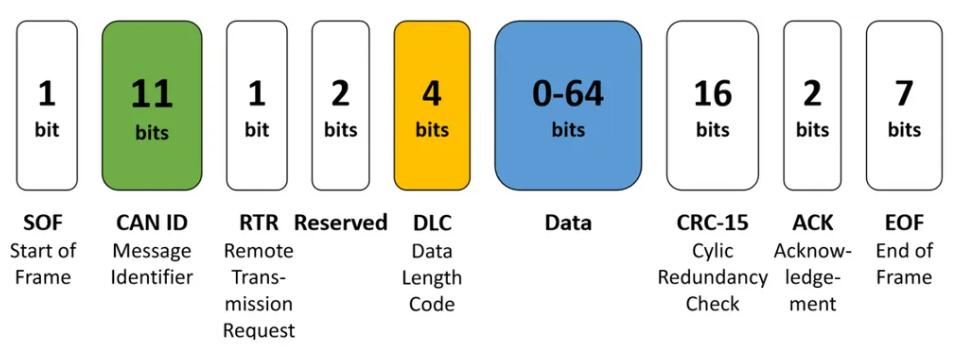
# **Vehicle Signal Specification(VSS) -- Advanced**

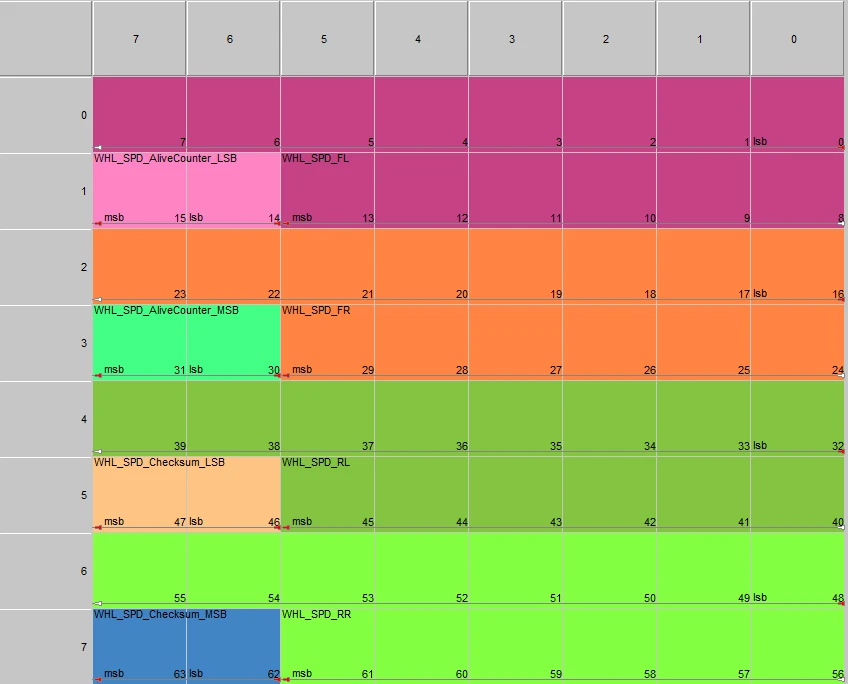
CAN

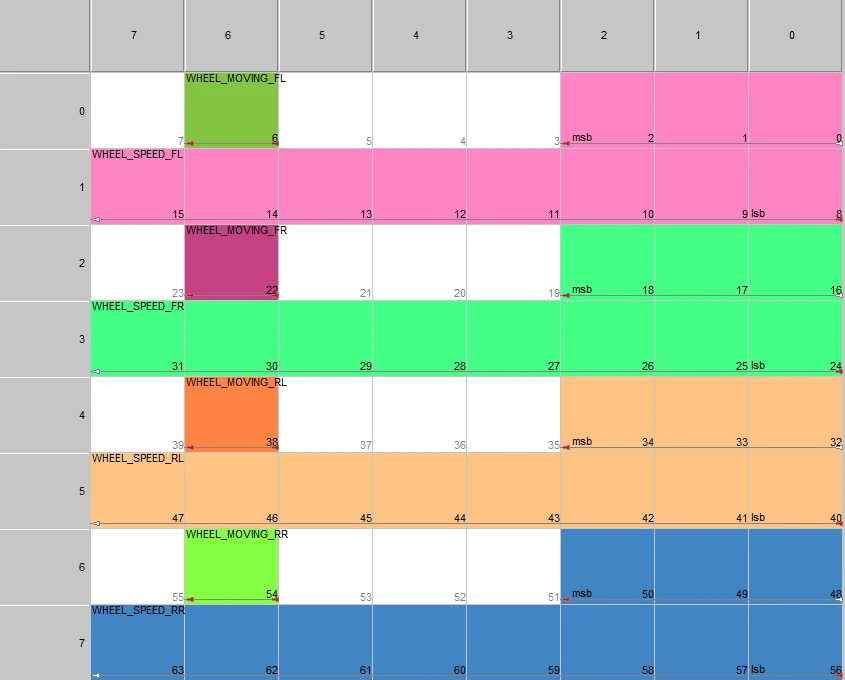


CAN Frame에서 데이터는 0 ~ 8 byte 까지만 사용

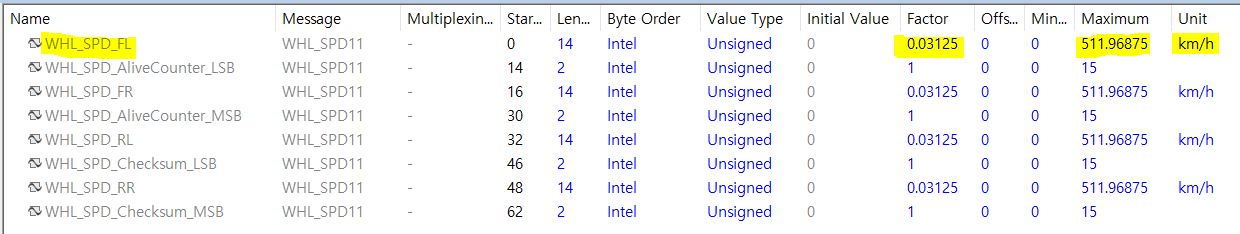
bit 단위로 잘라 데이터를 격납

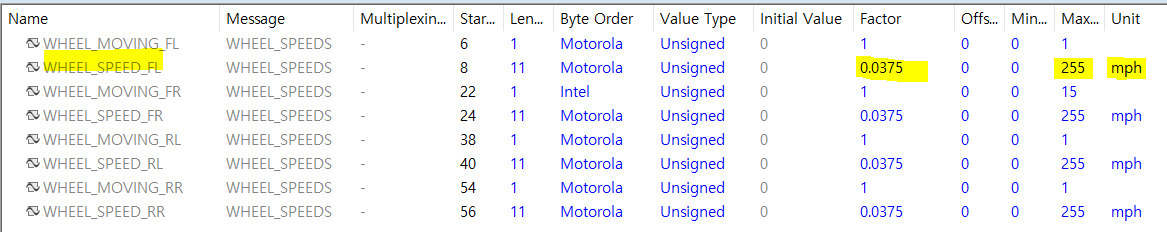
사용하는 CAN Frame은 아래와 같이 OEM별로 서로 다르다.





정보를 담는데 사용되는 시그널의 비트수도 차이가 나지만 아래와 같이 값을 해석하는 방법 또한 차이가 있다.





## **표준화된 시그널 표현 방식의 필요**

차량의 시그널을 (domain을 기반으로)분류하는 방법을 정의하는데 목적ㅁ

## **VSS (Vehicle Signal Specification)**

가장 큰(넓은) Domain인 Vehicle **“Node”**로 시작하여 하위 Domain으로 **“Branch”** node를 확장해 가면서 최종 Leaf node에서 특정 **“Signal”**이 정의되는 Tree 구조

* Branch : 다른 branch나 signal을 포함하는 node
* Signal(Data) : 특정한 값을 가질 수 있는 node
  + Sensor : 입력되는 값
  + Actuator : 출력되는 값
  + Attribute : 한번 값이 결정되면 변하지 않는 속성을 가진 시그널

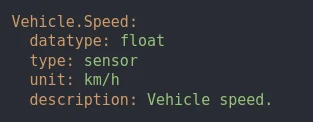
### **Data Types**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Min** | **Max** |
| uint8 | unsigned 8-bit integer | 0 | 255 |
| int8 | signed 8-bit integer | -128 | 127 |
| uint16 | unsigned 16-bit integer | 0 | 65535 |
| int16 | signed 16-bit integer | -32768 | 32767 |
| uint32 | unsigned 32-bit integer | 0 | 4294967295 |
| int32 | signed 32-bit integer | -2147483648 | 2147483647 |
| uint64 | unsigned 64-bit integer | 0 | 2^64 - 1 |
| int64 | signed 64-bit integer | -2^63 | 2^63 - 1 |
| boolean | boolean value | 0/false | 1/true |
| float | floating point number | -3.4e -38 | 3.4e 38 |
| double | double precision floating point number | -1.7e -300 | 1.7e 300 |
| string | character string | n/a | n/a |

### **Structs**

### **Data Unit Types**

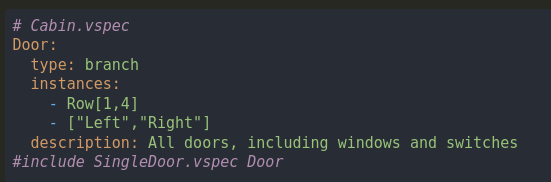
차량의 시그널은 해당 시그널에 담겨 있는 값이 어떤 데이터 타입인지가 매우 중요하기도 하지만 어떤 Unit으로 표현되는지도 매우 중요하다.



|  |  |  |
| --- | --- | --- |
| **Unit type** | **Domain** | **Description** |
| mm | Distance | Distance measured in millimeters |
| cm | Distance | Distance measured in centimeters |
| m | Distance | Distance measured in meters |
| km | Distance | Distance measured in kilometers |
| inch | Distance | Distance measured in inches |
| km/h | Speed | Speed measured in kilometers per hours |
| m/s | Speed | Speed measured in meters per second |
| m/s^2 | Acceleration | Acceleration measured in meters per second squared |
| cm/s^2 | Acceleration | Acceleration measured in centimeters per second squared |
| ml | Volume | Volume measured in milliliters |
| l | Volume | Volume measured in liters |
| cm^3 | Volume | Volume measured in cubic centimeters |
| celsius | Temperature | Temperature measured in degree celsius |
| degrees | Angle | Angle measured in degrees |
| degrees/s | Angular Speed | Angular speed measured in degrees per second |
| W | Power | Power measured in watts |
| kW | Power | Power measured in kilowatts |
| PS | Power | Power measured in horsepower |
| kWh | Energy Consumption | Energy consumption measured in kilowatt hours |
| g | Weight | Mass measured in grams |
| kg | Weight | Mass measured in kilograms |
| lbs | Weight | Mass measured in pounds |
| V | Electric Potential | Electric potential measured in volts |
| A | Electric Current | Electric current measured in amperes |
| Ah | Electric Charge | Electric charge measured in ampere hours |
| ms | Time | Time measured in milliseconds |
| s | Time | Time measured in seconds |
| min | Time | Time measured in minutes |
| h | Time | Time measured in hours |
| day | Time | Time measured in days |
| weeks | Time | Time measured in weeks |
| months | Time | Time measured in months |
| years | Time | Time measured in years |
| UNIX Timestamp | Time | Unix time is a system for describing a point in time. It is the number of seconds that have elapsed since the Unix epoch, excluding leap seconds. |
| mbar | Pressure | Pressure measured in millibars |
| Pa | Pressure | Pressure measured in pascal |
| kPa | Pressure | Pressure measured in kilopascal |
| stars | Rating | Rating measured in stars |
| g/s | Mass per time | Mass per time measured in grams per second |
| g/km | Mass per distance | Mass per distance measured in grams per kilometers |
| kWh/100km | Energy Consumption per distance | Energy consumption per distance measured in kilowatt hours per 100 kilometers |
| ml/100km | Volume per distance | Volume per distance measured in milliliters per 100 kilometers |
| l/100km | Volume per distance | Volume per distance measured in liters per 100 kilometers |
| l/h | Flow | Flow measured in liters per hour |
| mpg | Distance per Volume | Distance per volume measured in miles per gallon |
| N | Force | Force measured in newton |
| Nm | Torque | Torque measured in newton meters |
| rpm | Rotational Speed | Rotational speed measured in revolutions per minute |
| Hz | Frequency | Frequency measured in hertz |
| ratio | Relation | Relation measured as ratio |
| percent | Relation | Relation measured in percent |
| … | … |  |

## **Instances**

Branch에 아래와 같이 instance 속성을 정의하여 해당 branch가 가질 수 있는 경우의 수를 표현할 수 있다. Row 1~4 \* [left,right]



## **Includes**

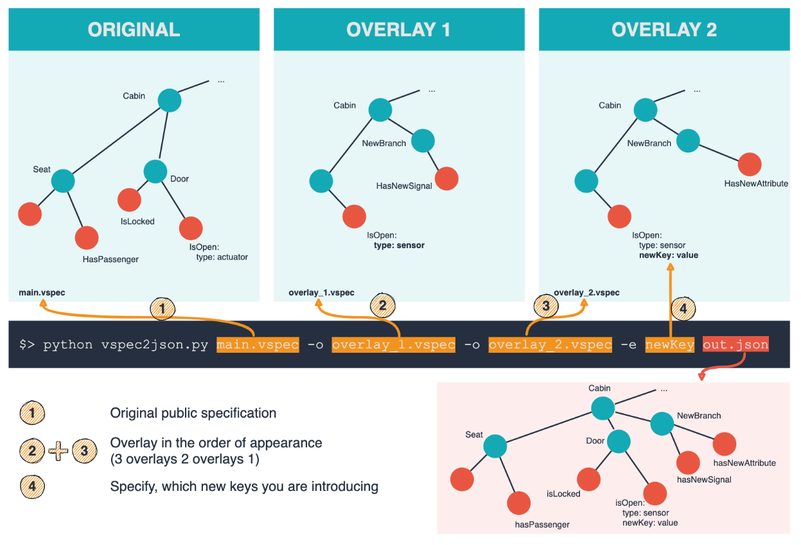
VSS는 vspec 이라는 확장자를 사용하여 시그널을 정의할 수 있다.

그런데, 이 vspec 파일은 “#include” 라는 명령을 사용하여 특정 vspec 파일에 다른 vspec 파일이 포함될 수 있도록 할 수 있다.

VSS는 vspec 이라는 확장자를 사용하여 시그널을 정의할 수 있다.

## **Overlay**

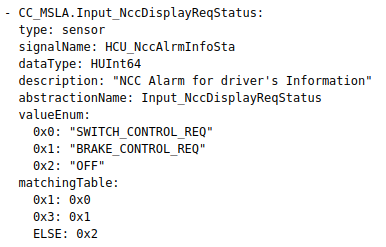
Overlay는 C++의 상속을 통한 확장 및 오버라이딩과 동일한 개념



## **VSM**

VSM은 HKMC가 VSS를 참조하여 기능을 확장하여 추상화된 차량의 시그널을 Application이 바로 사용할 수 있도록 그 기능을 제공

특정 Raw CAN Signal 과의 매칭 정보와 Raw CAN Value를 해석하는 방법을 정의할 수 있다



ValueEnum -> matchingTable로 해석방법을 정의할 수 있다는 것이 특징적.