

# DBC文件学习

---

参考如下:

视频参考:

文档参考:

[DBC文件学习](#)

[DBC文件到底是个啥](#)

[DBC文件解析](#)

## 学习进度

---

- ☐ (一)\*\*DBC概念:\*\* Database Can
- ☐ (二)win10下CANdb++编辑器的下载和安装
- ☐ (三)概念
- ☐ (四)使用candb++DBC文件的创建步骤
  - []

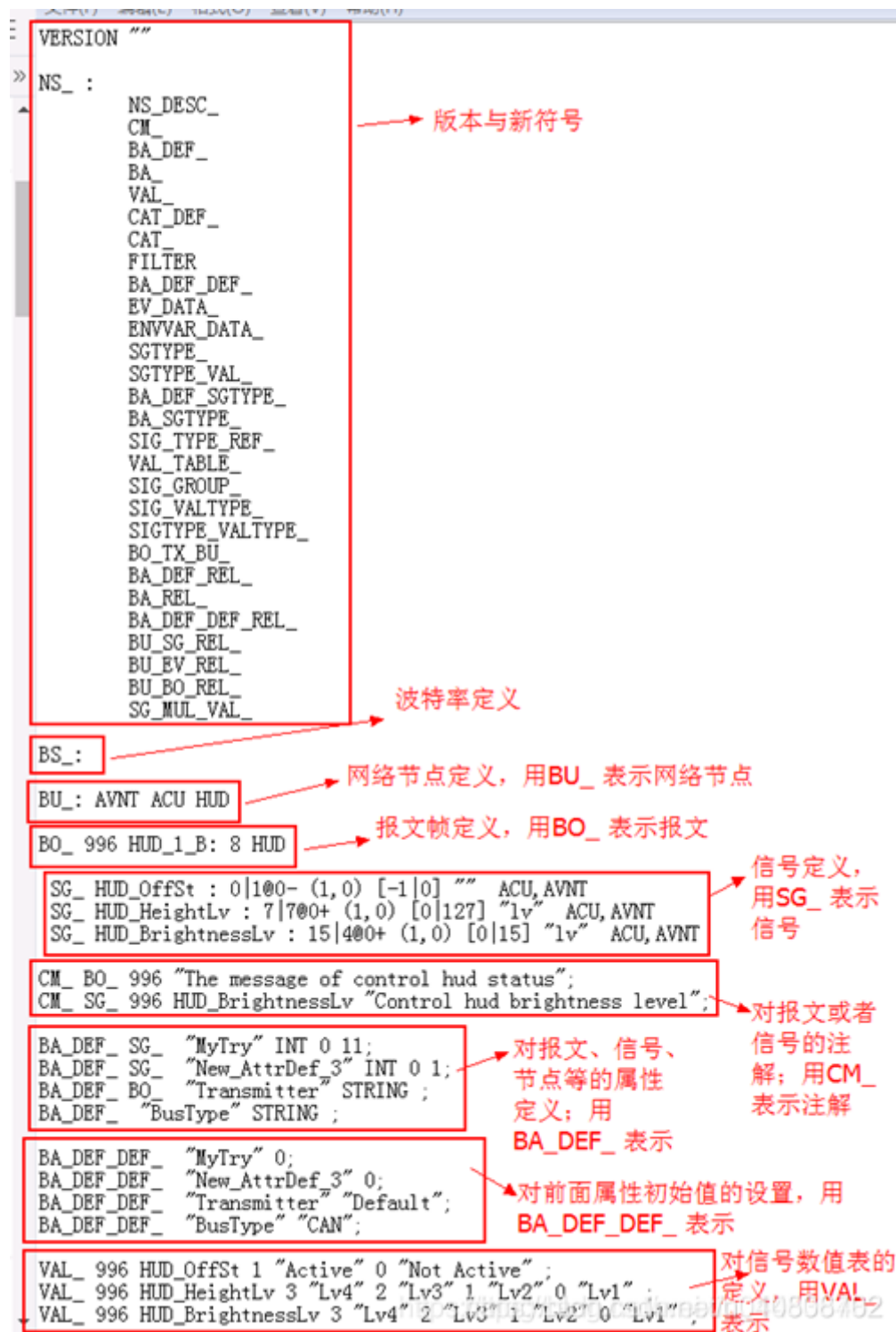
---

### (一) \*\*DBC概念:\*\* Database Can

dbc数据库文件是一种用来描述CAN总线协议中协议数据及其代表的具体意义, 可以用来监测与分析CAN网络上的报文数据, 也可以用来模拟某个CAN节点。DBC文件是以Tag (标签) 来标识每一个元素。

### (二) [win10下CANdb++编辑器的下载和安装](#)

### (三) 概念



1、如下图, Dbc文件头部包含着"version"与"new symbol"的信息; (1)、"version"信息可以为空, 也可以由用户自定义; (2)、"new symbol"信息在我们创建dbc文件时就已经自动生成: 所以这一部分的信息我们无需过多留意, 一般默认即可。

## 2、波特率定义

BS\_: [baudrate: BTR1, BTR2]; 其中BS\_为关键字, 用于定义CAN网络的波特率; [ ]内容表示为可选部分, 可以省略 (如下图例子中即把该部分省略了); 但关键字"BS\_"必须存在, 省略则会出错。《DBC File Format Document》规范中明确提醒, 必须保留BS\_标签。

3、网络节点的定义 格式如下： BU\_ : Nodename1 Nodename2 Nodename3 .....

解释： BU\_ 为关键字，表示网络节点 Nodename1、Nodename2 网络节点名字，由用户自己定义；

注意事项： 需要保证节点命名的唯一性

4、报文帧的定义 格式如下： BO\_ MessageId (10进制数表示) MessageName: MessageSize Transmitter 解释： 1)、BO\_ 为关键字，表示报文；

2)、MessageId 报文ID，是以10进制数表示的；（如例子中的996，代表报文ID为0x3E4，是longlong类型，也就是CAN ID的值；）

3)、MessageName 报文的名称，命名规则和C语言变量相同；

4)、MessageSize 报文数据域字节数，为无符号整型数据，CAN 2.0为最大8字节，CAN FD 最大64字节；

5)、Transmitter 该报文的网络节点；如果该报文没有指定发送节点，则该值需设置为“Vector\_XXX”。

如示例中的BO\_ 201959408 MSG1: 8 VCU\_Vehicle\_Control\_Unit 报文ID: 201959408

(0xC09A7F0) 报文名: MSG1 分隔符: “:” 报文长度: 8字节 报文发送者:

VCU\_Vehicle\_Control\_Unit (由VCU\_Vehicle\_Control\_Unit这个节点发出的，数据域长度为8字节，ID为201959408 (0xC09A7F0)，名字命名为MSG1的报文)

5、信号的定义 格式如下：

SG\_ SignalName (SigTypeDefintion) : StartBit|SignalSize@ByteOrder ValueType  
(Factor,Offset) [Min|Max] Unit Receiver

解释：

1)、SG\_ 为关键字，表示信号；

2)、SignalName(SigTypeDefintion) : 表示该信号的名字 和 多路选择信号的定义；

2.1) SigTypeDefintion是可选项，有3种格式： a) 空，表示普通信号。 b) M，表示多路选择器信号。 c) m50，表示被多路选择器选择的信号，50，表示当M定义的信号的值等于50的时候，该报文使用此通路。

3)、StartBit、SignalSize 表示该信号起始位、信号长度；

4)、ByteOrder 表示信号的字节顺序：0代表Motorola格式，1代表Inter格式；

5)、ValueType 表示该信号的数值类型：+表示无符号数，-表示有符号数；

6)、Factor, Offset 表示因子, 偏移量; 这两个值用于信号的原始值与物理值之间的转换。

转换如下: 物理值=原始值\*因子+偏移量;

7)、Min|Max 表示该信号的最小值和最大值, 即指定了该信号值的范围; 这两个值为double类型;

8)、Unit 表示该信号的物理单位, 为字符串类型;

9)、Receiver 表示该信号的接收节点; 若该信号没有指定的接收节点, 则必须设置为"Vector\_XXX"。

如示下面的例中: 第一个信号: SG\_S\_Check: 45|10@0+ (0.00625,0) [0|160] "M" EL3160\_60,ESC 表示定义了一个命名为 S\_Check的普通信号, 其起始位是第45位, 信号长度10 bit; 信号是Motorola格式, 数值类型为无符号类型数; 因子为0.00625, 偏移量为0; 信号取值范围为0到160; 信号物理单位为字符串"M"; 该信号接收节点为EL3160\_60,ESC这两个节点。、第二个信号: SG\_Send\_Mux M: 7|8@0+ (1,0) [0|0] "" EL3160\_60,ESC 表示定义了一个命名为 Send\_Mux的多路选择器信号, 其起始位是第7位, 信号长度8 bit; 信号是Motorola格式, 数值类型为无符号类型数; 因子为1, 偏移量为0; 信号取值范围为0到0; 信号物理单位为字符串""; 该信号接收节点为EL3160\_60,ESC这两个节点。该信号做选择通道使用。、第三个信号: SG\_S\_Level\_A\_Voltage m50: 15|16@0+ (0.00625,0) [0|160] "V" EL3160\_60,ESC 表示定义了一个命名为S\_Level\_A\_Voltage的被选择信号, 其起始位是第15位, 信号长度16 bit; 信号是Motorola格式, 数值类型为无符号类型数; 因子为0.00625, 偏移量为0; 信号取值范围为0到160; 信号物理单位为字符串"V"; 该信号接收节点为EL3160\_60,ESC这两个节点。

6、注解部分 格式如下:

CM\_ Object MessageId/NodeName "Comment"

解释: 1)、CM\_ 为关键字, 表示注解信息;

2)、Object 表示进行注解的对象类型, 可以是节点"BU\_"、报文"BO\_"、消息"SG\_";

3)、MessageId/NodeName 表示进行注解的对象, 若前面的对象类型是信号或者报文, 则这里的值应为报文的ID (10进制数表示); 若前面的对象类型为节点, 则这里的值应为节点的名字;

4)、Comment 表示进行注解的文本信息;

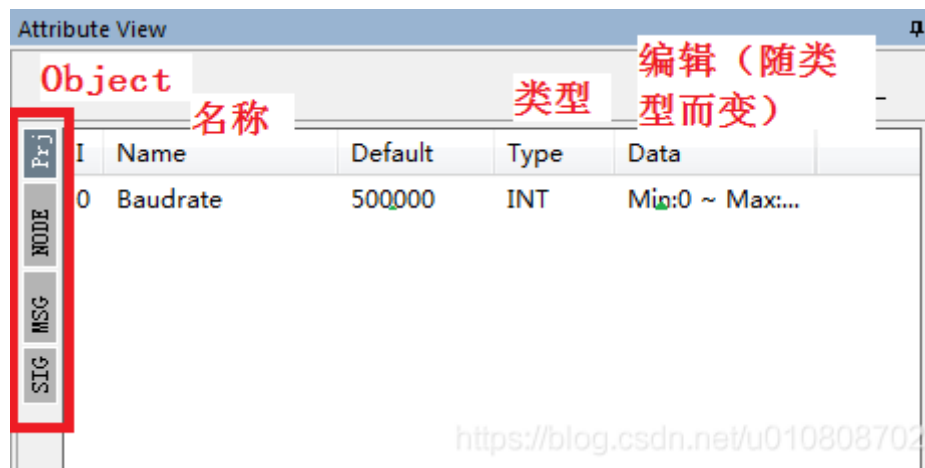
如示例中的 CM\_SG\_996 HUD\_HeightLv "Control hud height level";

表示对ID为996(0x3E4)这条报文下的名为"HUD\_HeightLv"的信号进行注解说明, 说明的内容为"Control hud height level"。

又如 CM\_BU\_HUD "Head Up Display";表示对HUD这个节点进行注解说明, 说明的内容为"Head Up Display"。

注释以“”包围，注释内部不允许出现“号。

7、特征（属性）定义部分：和特征相关的Tag一共有三条：1) BA\_DEF\_ 特征名称类型定义。格式如下：BA\_DEF\_ Object AttributeName ValueType Min Max; 解释：BA\_DEF 标签 Object 特征类型，可以是BU\_（节点特征定义）、BO\_（报文特征定义）、SG\_（信号特征定义）、空格（项目特征定义）；AttributeName 特征名称（C语言变量格式）ValueType 特征值类型（只能是十进制、十六进制、浮点数、枚举、字符5种类型）Min Max 数值类型这里出现范围，枚举类型这里是枚举值，字符类型，这里是空。



## 2) BA\_DEF\_DEF\_

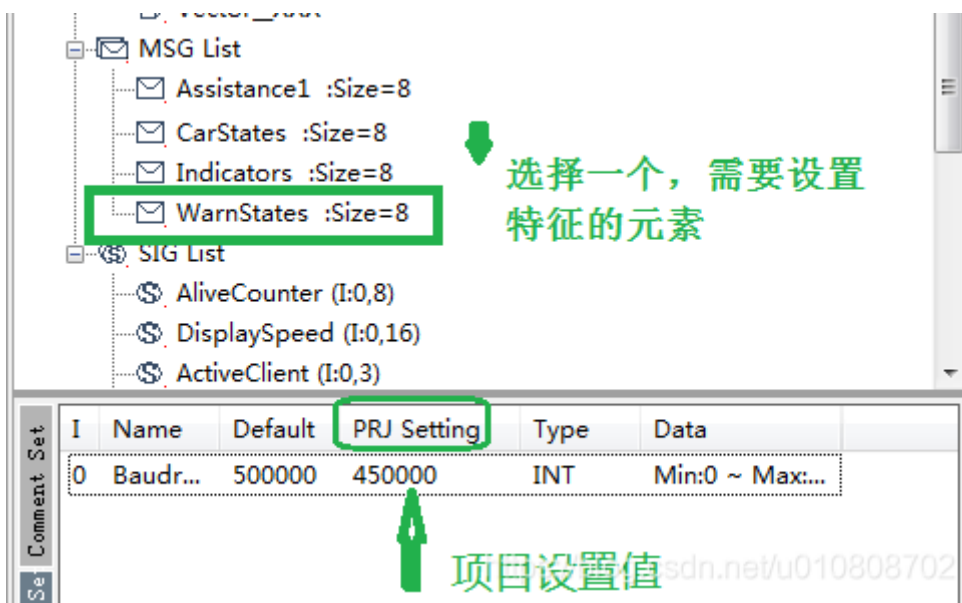
特征默认值定义。格式如下：BA\_DEF\_DEF\_ AttributeName DefaultValue; 解释：BA\_DEF\_DEF\_ 标签 AttributeName 特征名称（C语言变量格式）DefaultValue 该特征的默认设置值



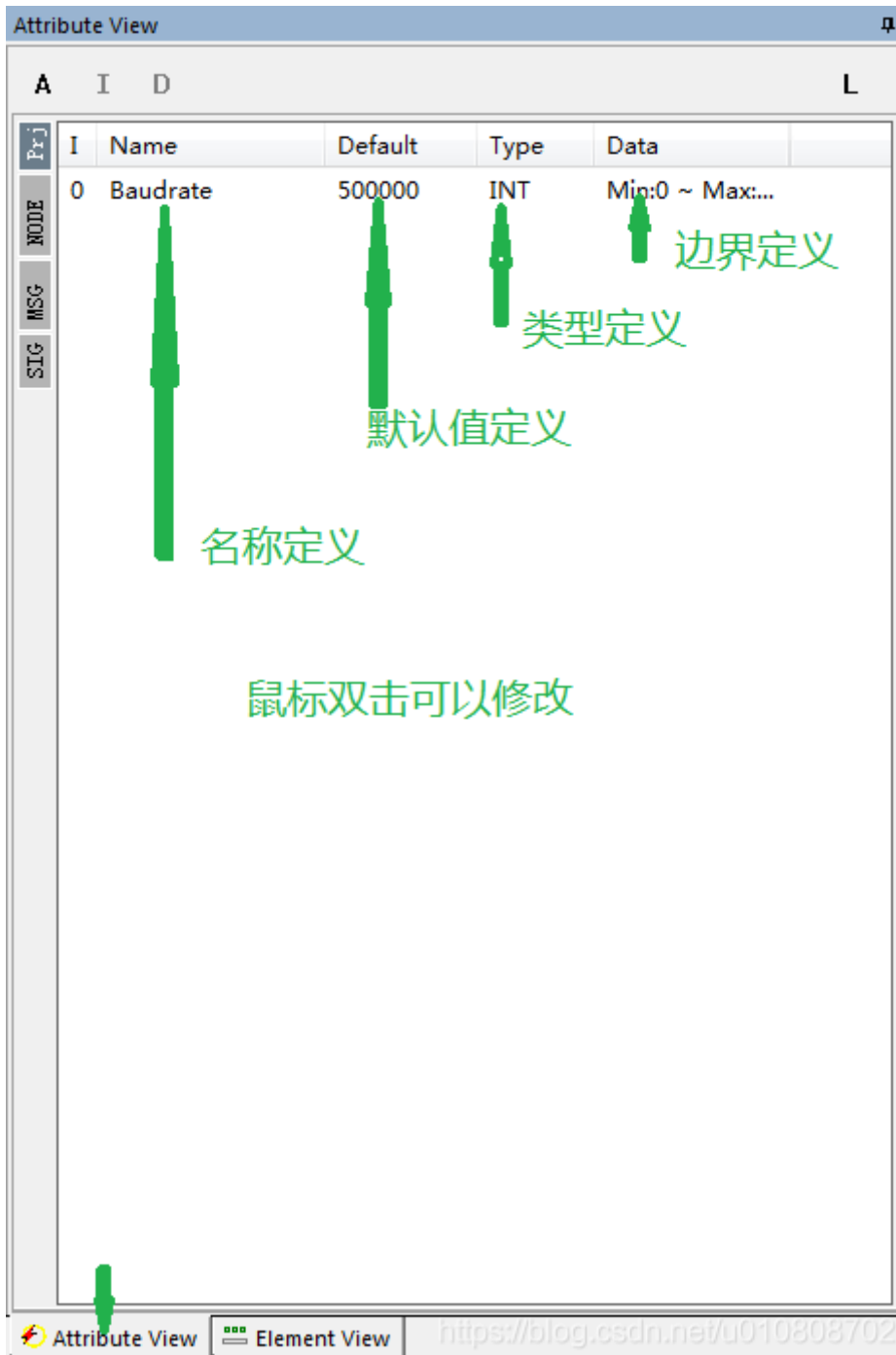
## 3) BA\_

特征项目设置值定义，格式如下：BA\_ AttributeName projectValue;

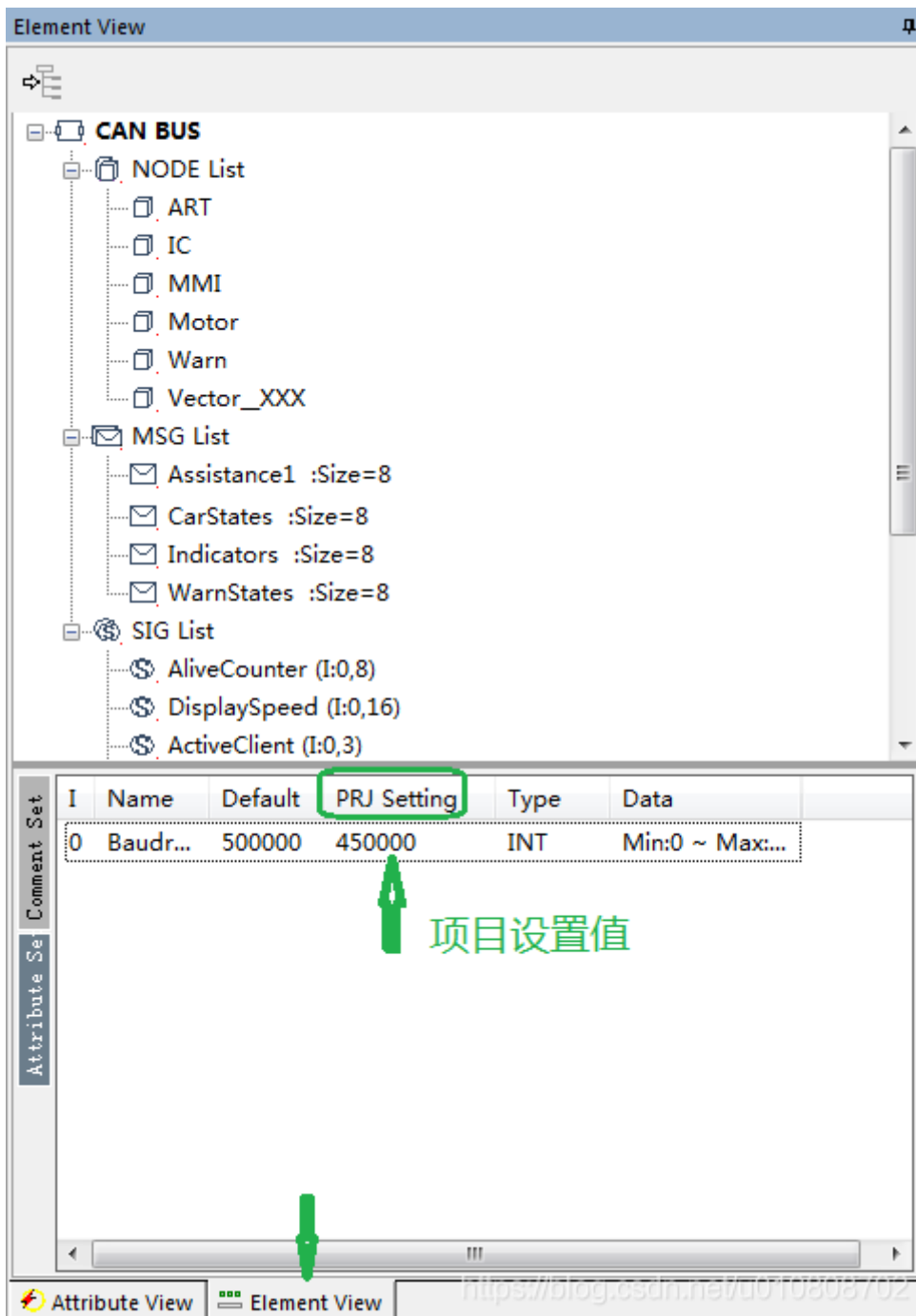
解释：BA\_ 标签 AttributeName 特征名称（C语言变量格式）projectValue 该特征的设置值



在DBCView4.3里面是把BA\_DEF\_ 和 BA\_DEF\_DEF\_合并在一起了，用了特征视图来管理，如下图。



然后把 特征项目设置值定义 BA\_AttributeName projectValue; 和元素合并在一起了。



8、数值表部分 格式如下： VAL\_ MessageId SignalName N "DefineN" ..... 0 "Define0"; 解释：（1）、VAL\_ 为关键字，表示数值表定义；

（2）、MessageId 表示该信号所属的报文ID（10进制数表示）；

（3）、SignalName 表示信号名；

（4）、N "DefineN" ..... 0 "Define0" 表示定义的数值表内容，即该信号的有效值分别用什么符号表示。

如示例中的 VAL\_ 996 HUD\_OffSt 1 "Active" 0 "Not Active";

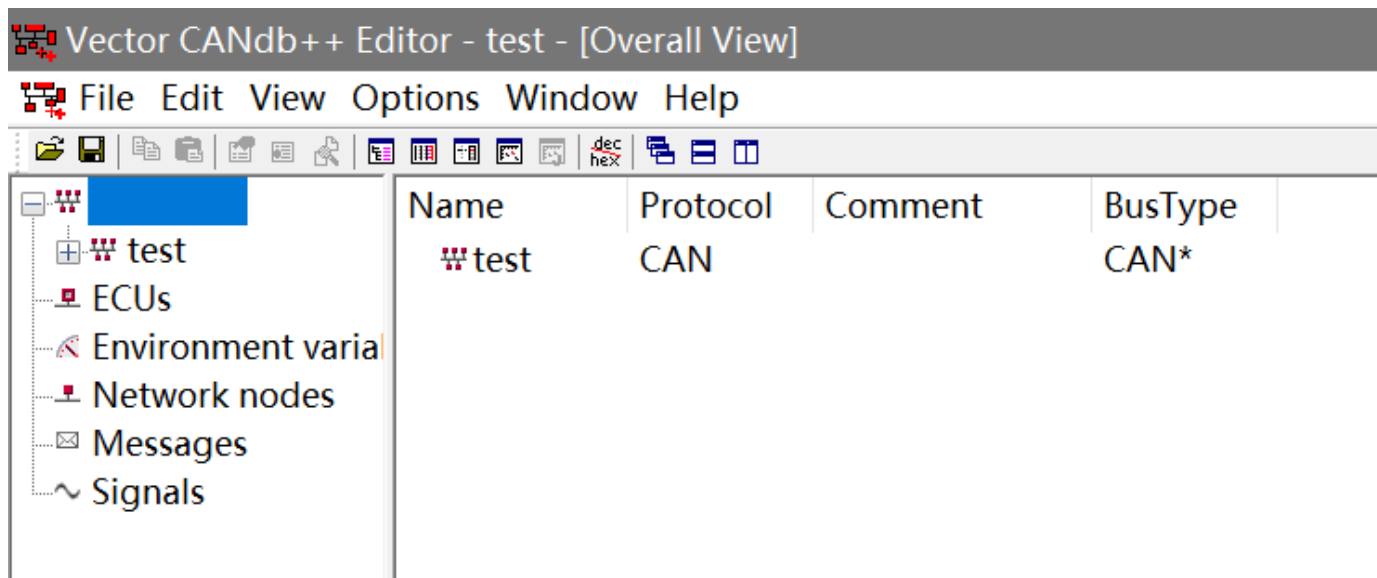
表示对ID为996(0x3E4)的这条报文下的，一个命名为"HUD\_OffSt"的信号，进行其数值表的定义；用"Active"取代1；用"Not Active"取代0。只有自然数类型的信号才可以使用数值表表示。



## (四) DBC文件的创建步骤:

1.创建一个CAN数据库文件 2.创建信息需要用到的数值表 3.创建信号并且关联数值表 4.创建报文 5.创建网络节点 6.将信号、报文及网络节点进行关联链接 7.创建或导入自定义属性并进行修改 8.一致性检查

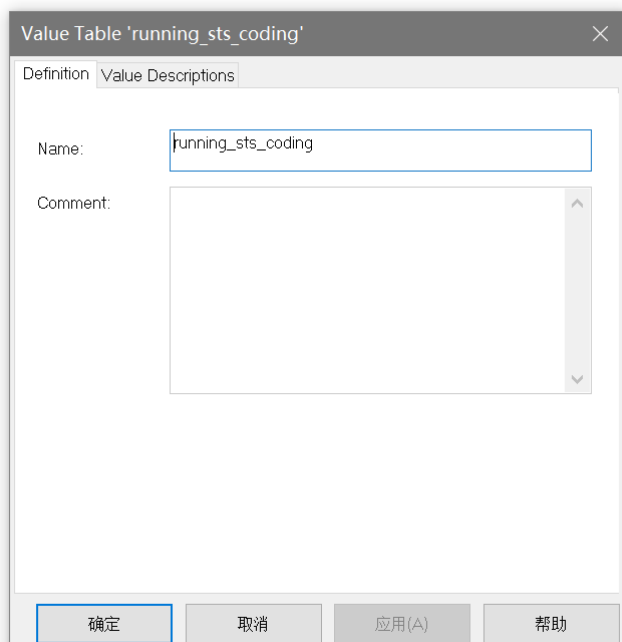
### 1.创建一个CAN数据库文件



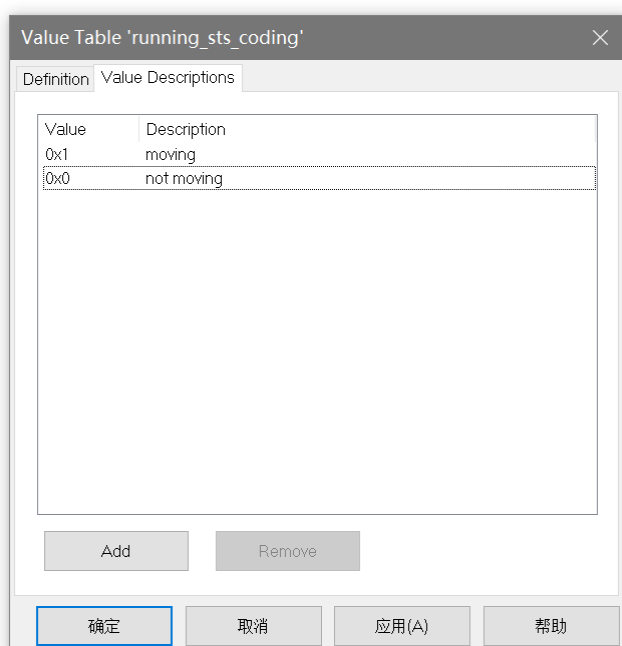
### 2.创建信息需要用到的数值表

创建数值表的意义是为了给后续创建的信号提供解释

创建数值表需要在数值表的视图中操作，通过主菜单的 "view"可以将视图切换到"value table"界面，打开value table界面之后，在空白处右键选择"new"即可新建。

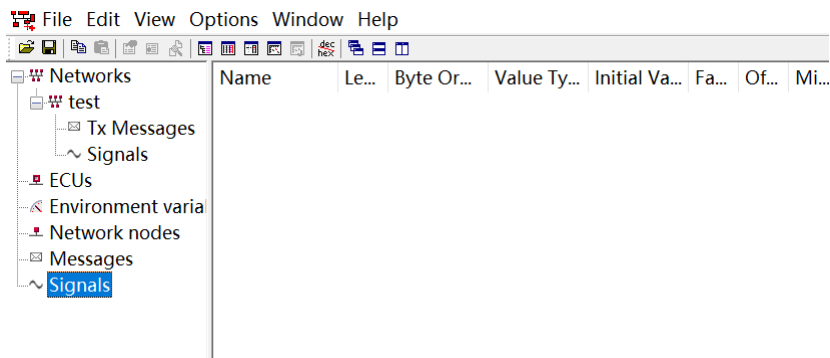


比如这里我们定义名称为 "running\_sts\_coding", 然后再value description中做一下定义, 如下:

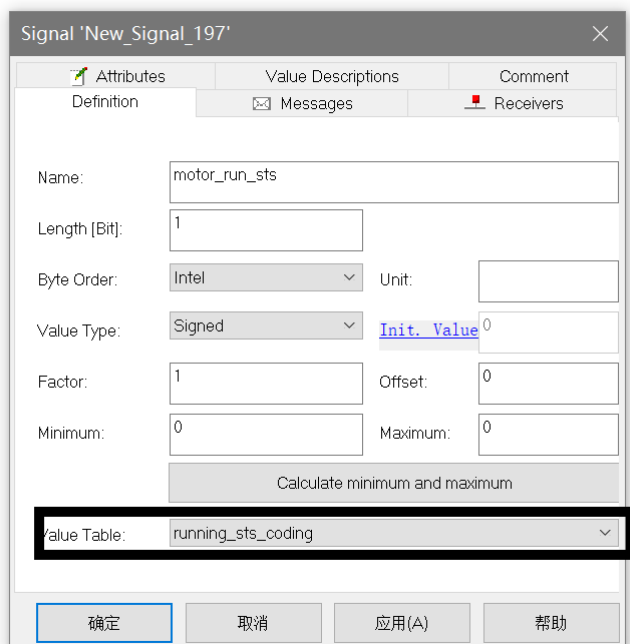


### 3.创建signal, 关联相应的数值表

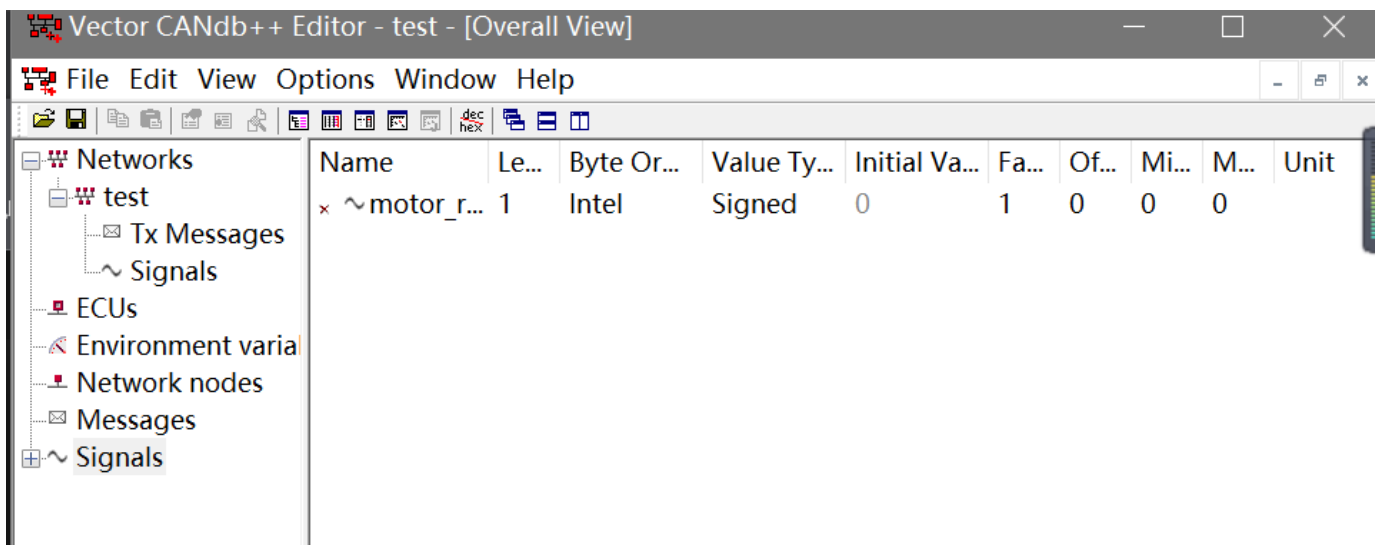
前面提到的创建数值表只是一个准备, 真正有意义的是信号, 而数值表就是为信号服务的, 因为数值表中对数值含义的解释可以完成对信号含义的解释。 通过菜单 "view" 将视图切换为 "overview", 如下图所示



然后在signals的位置点击右键，新建，打开如下界面，并做定义



关键一步，在value table 的位置选择我们在上一步建立的数值表，这样就将信号和数值表链接起来了，同时也完成了信号的创建。到此，点击确定即可。可以看到，在视图中新增了一个刚刚建立的信号



#### 4.创建message报文

在之前介绍CAN通讯的文章中我们说到，CAN通讯的载体是帧，也就是消息，而不是单纯的一个一个的信号，是把很多的信号封装到消息帧里面以帧的格式进行传输的，所以在建立了signal之后还需要将信号封装到帧中，那么就需要首先创建message，方法很简单，还是在“overview”的视图中，message位置点击右键，新建，定义如下

Message 'New\_Message\_25 (0x0)'

Layout | Attributes | Comment

Definition | Signals | Transmitters | Receivers

Name: motor

Type: CAN Standard

ID: 0x573 DLC: 8

Transmitter: No Transmitter

Tx Method: NoMsgSendType

Cycle Time: 0

确定 取消 应用(A) 帮助

然后，在第二个子选项卡中关联帧的信号

Choose Objects

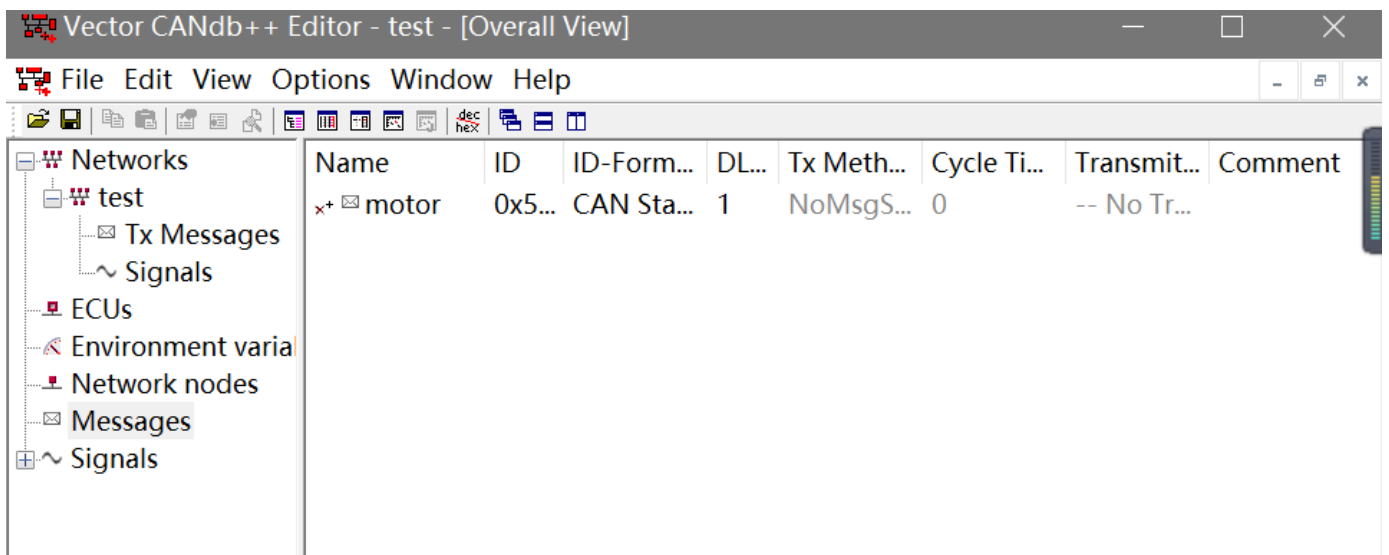
Filter by: Name

Value: motor\_run\_sts Filter

Name	Le...	Byte Order	Value Type
~motor_run_sts	1	Intel	Signed

OK Cancel Help

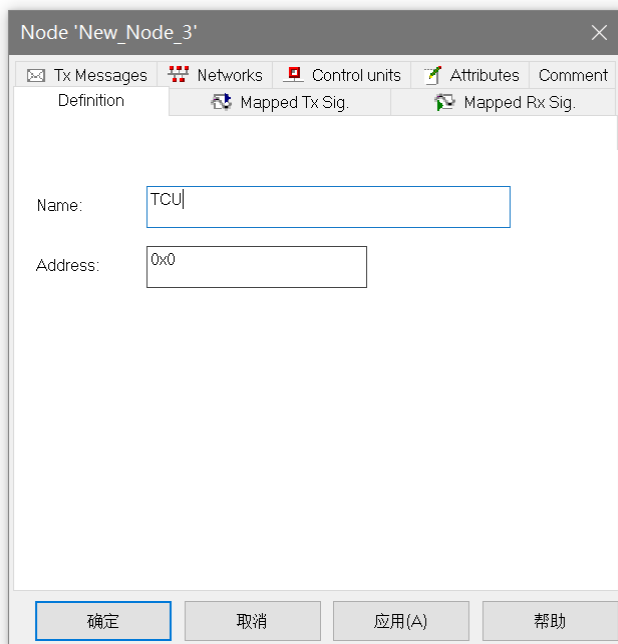
然后就可以确定了，定义之后在message界面就会出现一帧新的消息



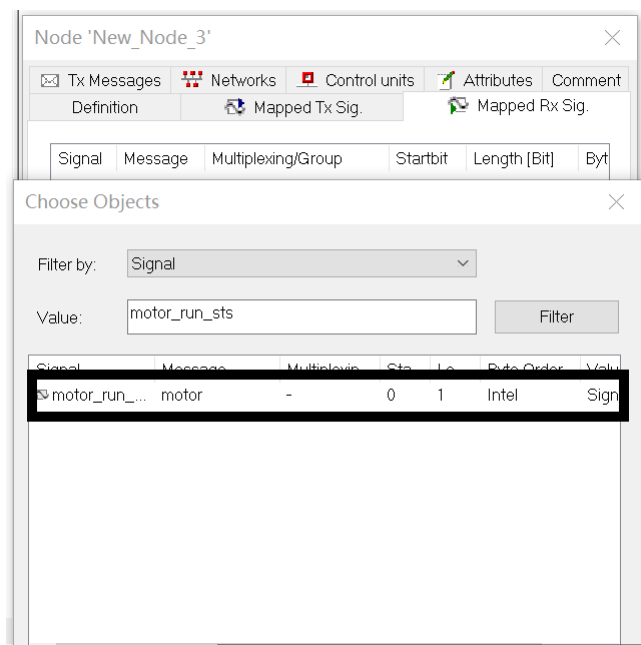
## 5.创建网络节点

这个节点也是需要进行定义的，方法同样很简单，在"overview"界面内的network处，点击右键，选择新建

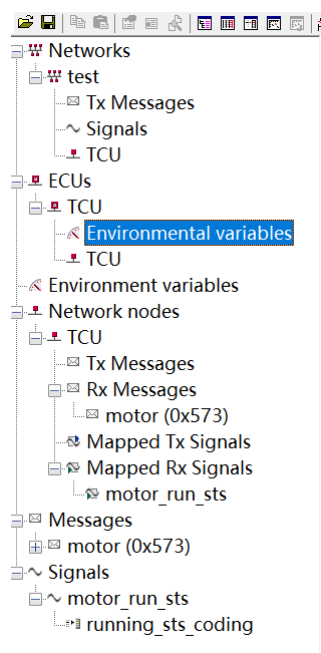
这里可以定义节点名称和节点地址，这里的地址有别于前面message中涉及的ID，这是两个不同的概念。比如，我们命名如下



然后在Tx message选项卡中设置发送的消息，因为这是第一个节点还有可接收的消息，所以只能设置发送的消息，我们只有一帧消息，所以设置如下：



## 小结



另外一点需要注意，如果是创建DBC文件，再创建完成之后一定要做一致性检查，菜单内file下的consistency check