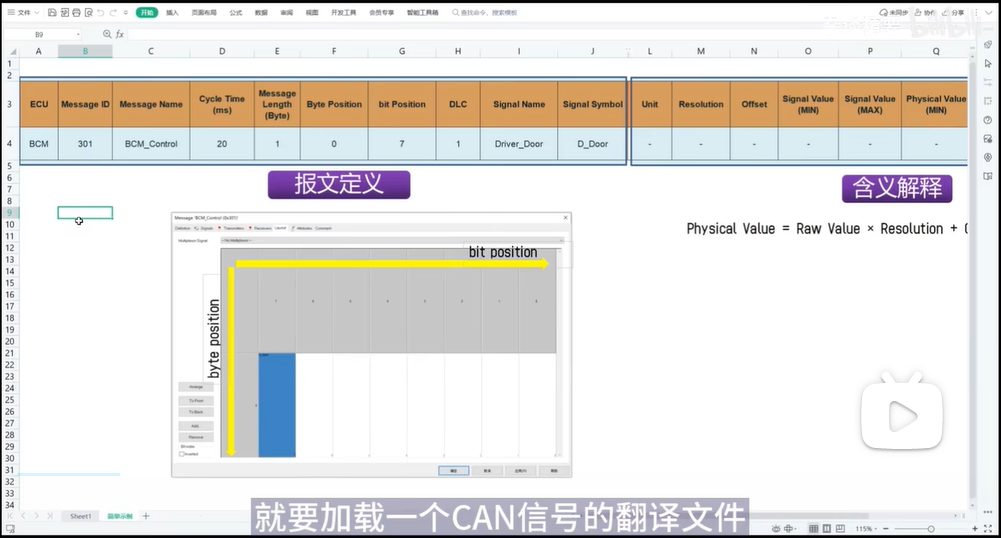
# DBC制作流程

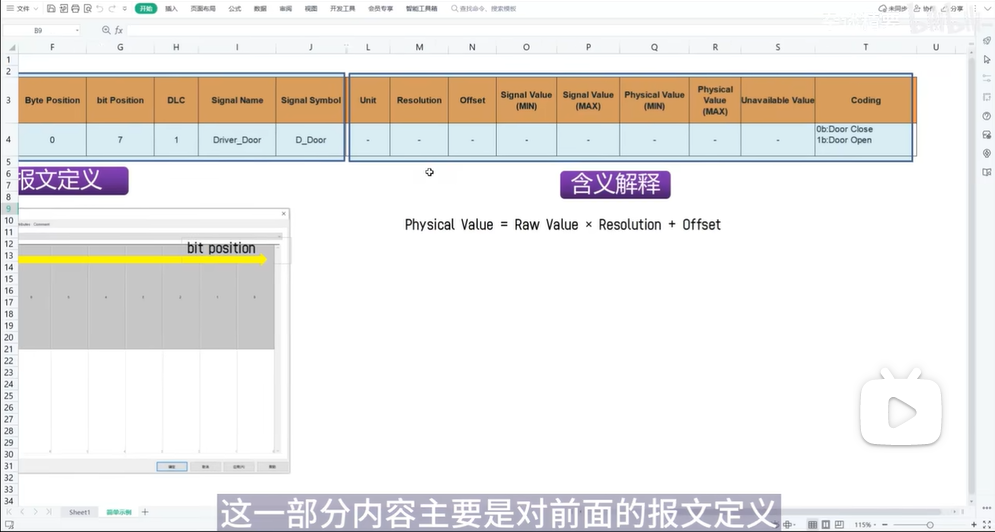


要用CANoe实时观察CAN信号的变化，就要加载一个CAN信号文件，DBC数据库。而在做dbc数据库之前，出了定义网络拓扑，还要定义每个ECU收发报文的内容及含义。上图中该表为CAN矩阵开发中进场使用的列表，它主要由两部分组成。一个是报文定义，一个是含义解释。报文定义部分包括ECU的名称，发送报文的ID，发送报文的名称，发送周期，字节长度。然后Byte position和Bit position是字节位置和数据位置的意思。

字节位置：

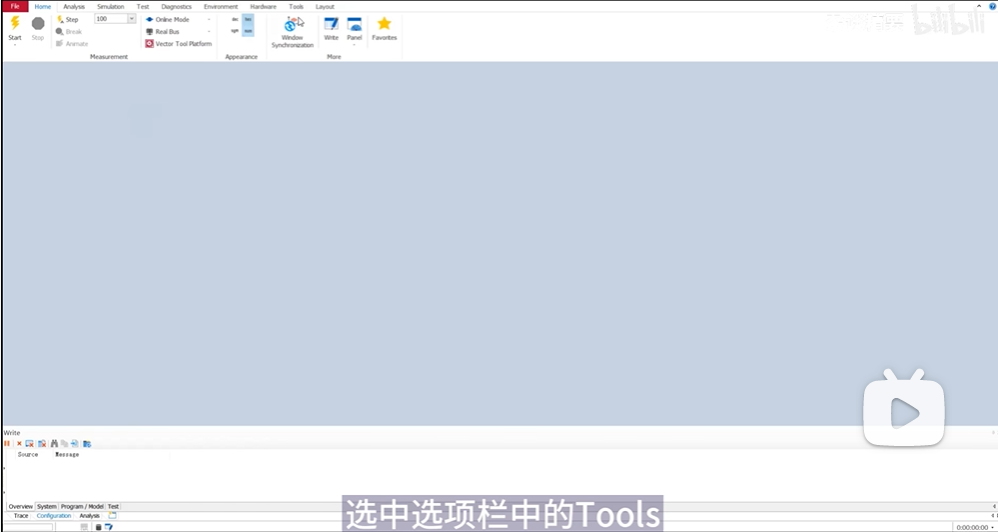
数据位置：

这两个值其实就是信号的坐标，CANoe将64位数据场划分为8\*8，64个小格子，字节纵向排列，字节中每一位信号横向排列。所以这里第0个byte就是第一行，第七个位就是第一行第一个数。这里byte position从上到下分别是第0到第7个字节，而bit position从左到右是第7位到第0位。DLC是数据长度，这里为1，就表示驾驶车门信号的状态就只有1位数据，也就是0和1两种状态。

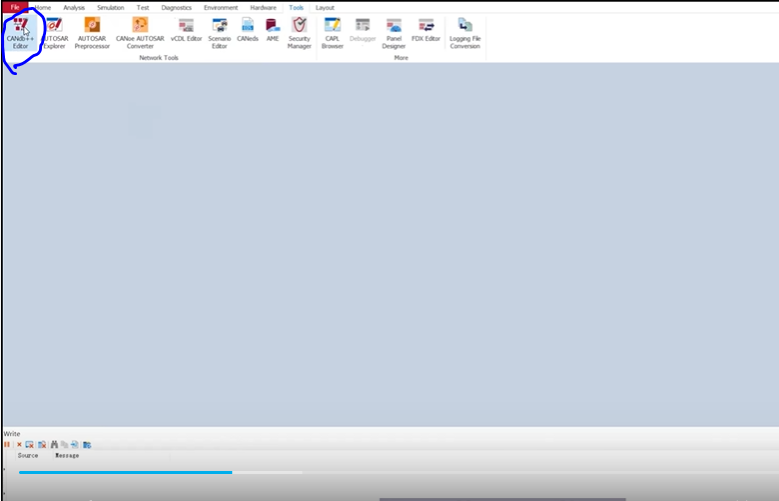


接下来看下第二部分含义的内容解释，这一部分内容主要是对前面的报文定义进行换算或者翻译，Unit是单位，Resolution是系数，Offset是偏置数。刚才矩阵中的原始值要乘上，系数然后加上偏置数才是最终的物理值。Signal value最大最小值是指信号原始值，Physical Value就是换算完以后的物理值，Unavailable Value是当控制单元发生故障或者信号有误时，发送的默认值，表示此时信号出错，告诉其它控制单元不要采用，Coding是对信号状态的翻译内容，这里门的0/1信号就分别代表门的关闭和打开。

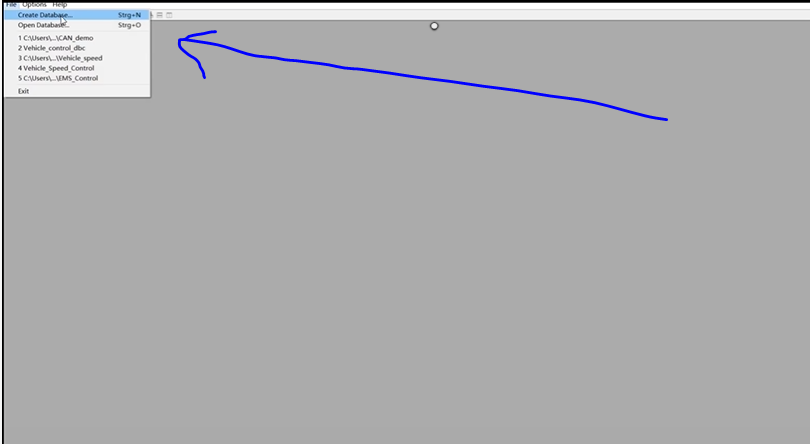
接下来打开CANoe来制作dbc数据库，



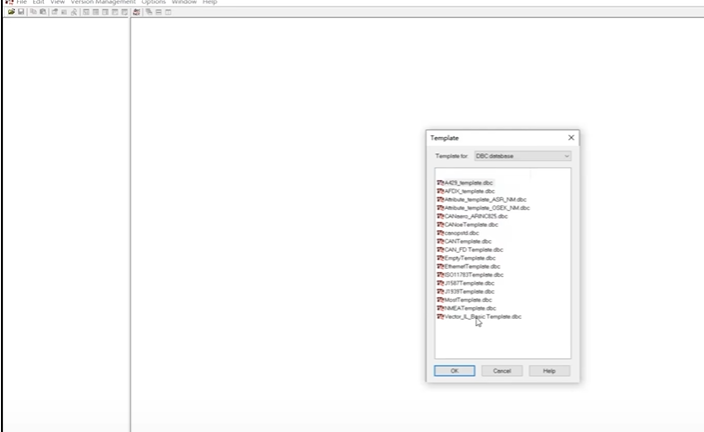
选中选项栏中的Tools Candb++ editor

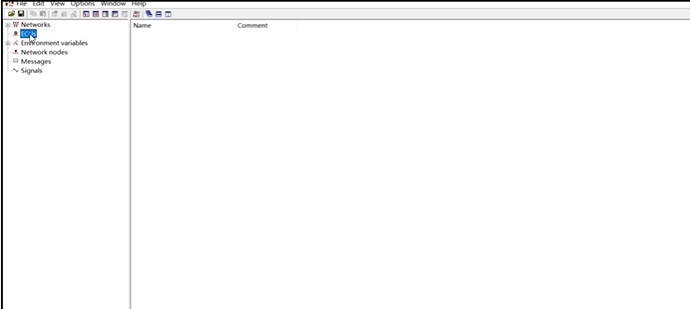


然后选中File Create database

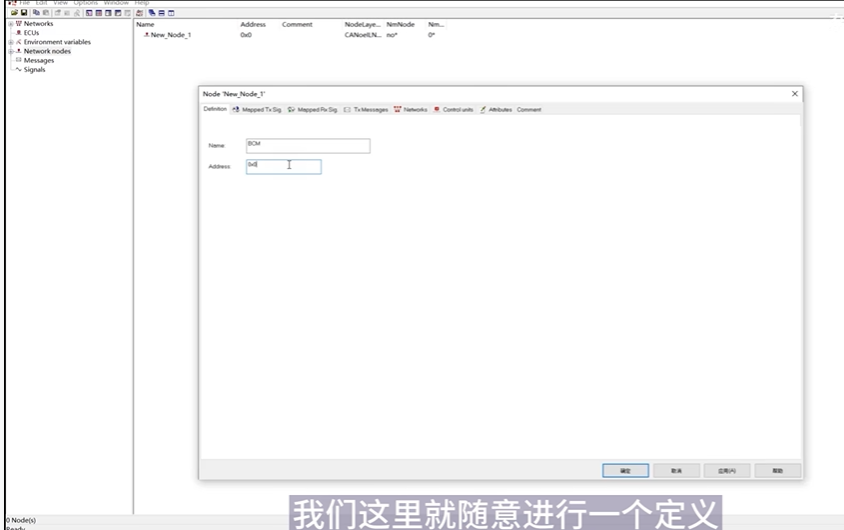


建立数据库，这里有很多通信协议的模板，一般选择CAN template dbc，或者是Vector公司自带的一个基础模板，这里选择Basic Template。





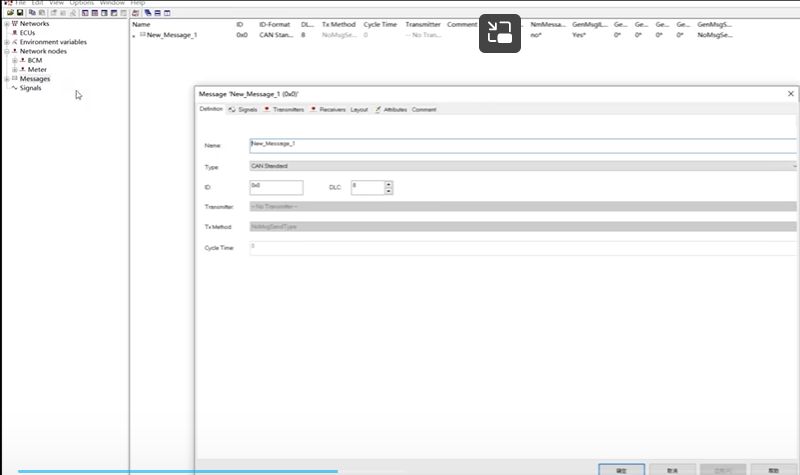
保存后可以看到左边的信息栏中出现了6个分类，network是我们这一次建立示例的网络，ECUS是ECU，ECU就代表你要仿真的这个控制器，这个控制器在DBC数据库里面是不可以进行创建的。接下来在网络节点创建完成以后，会自动生成ECU，Environment Variables是指环境变量，环境变量的创建，稍后说。Network Node就是指你要仿真的网络节点，此次仿真节点主要有两个，一个是仪表，一个是BCM。在Network node下新建结点BCM，然后输入其网络地址，实际开发中会有固定地址的一个代码，



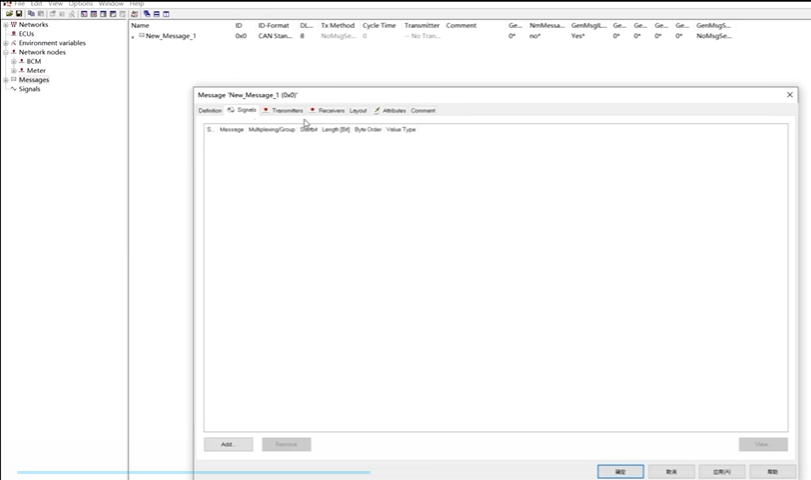
这里随意定义一个数值后，点击确定，可以看到网络节点下面有一个BCM



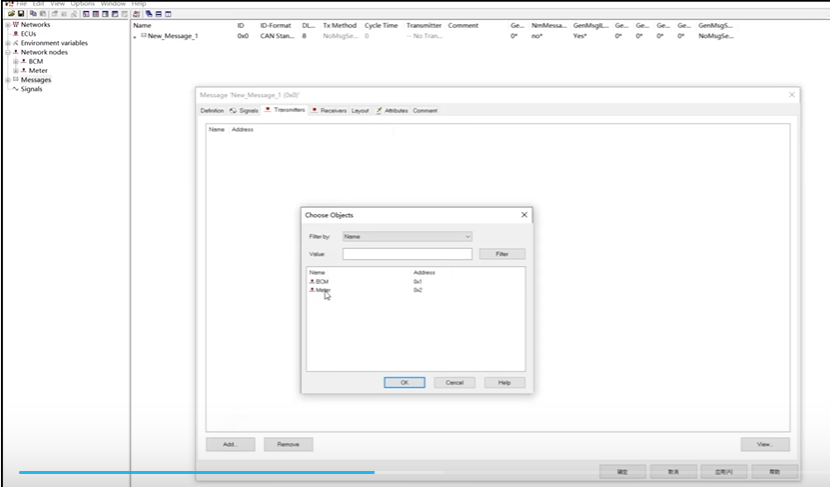
然后新建仪表，有了网络节点以后，接着进行报文的添加，



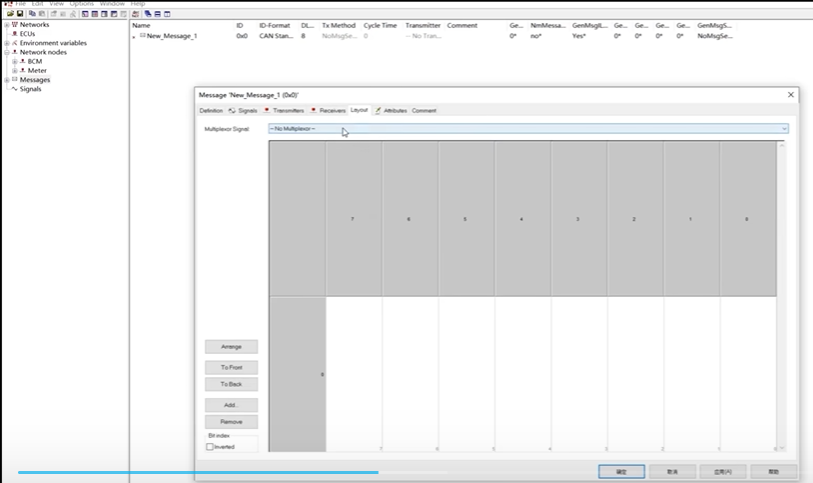
这里可以对照刚才Excel中报文的定义进行填写，报文名称填为BCM\_Control,选择标准的CAN格式，如果是扩展格式的话，也可以选择Extended，然后ID是301，数据长度是只有一个字节，



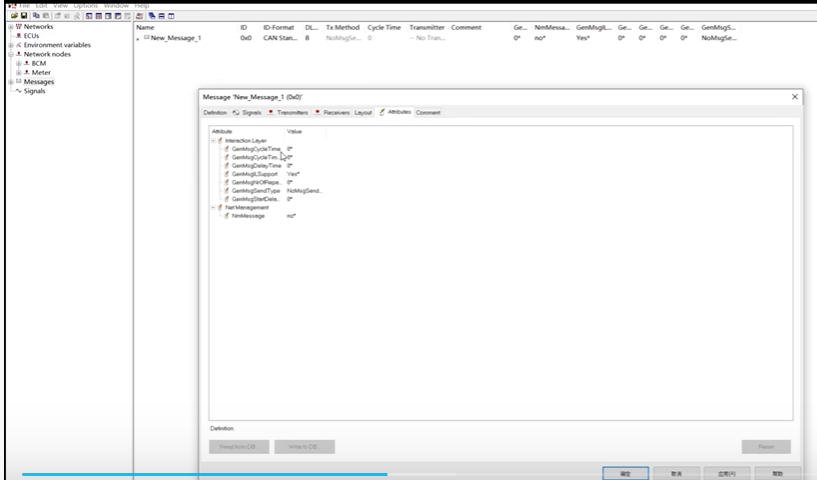
Signal的话等创建完信号再进行添加，



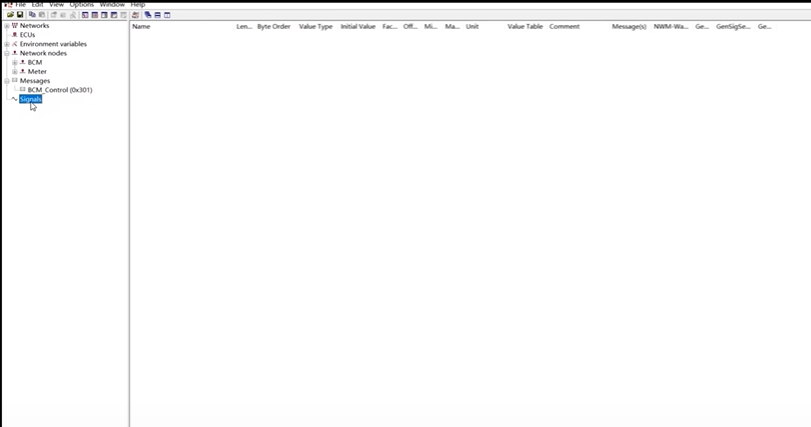
Transmitter的话，发送这条报文的节点可以选择BCM，这里接受节点Receiver的话可以不用填写，layout也是再信号添加进来以后进行调整



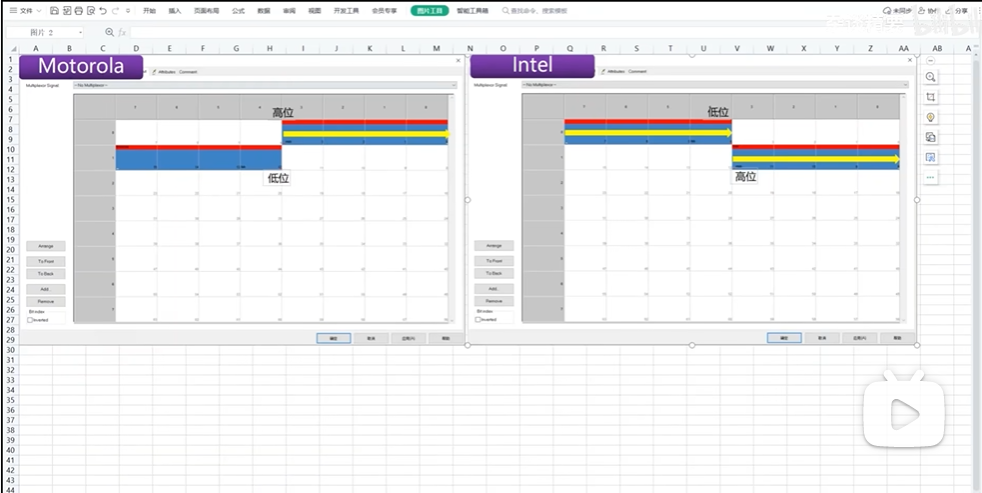
在Attribute中，需要选择报文的发送周期，以及发送方式



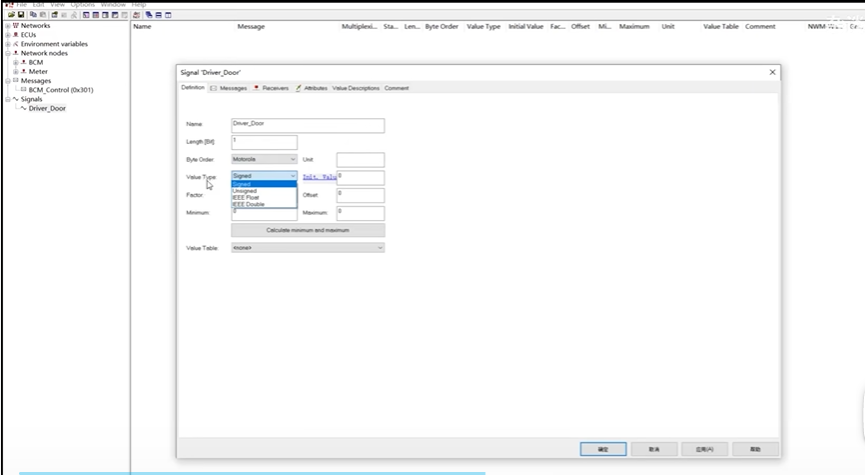
这里Message cycle time选择20ms。Message send type选择cyclic周期性发送，确定后，可以在Messages菜单栏下看到BCM Control这条报文已经添加进来，



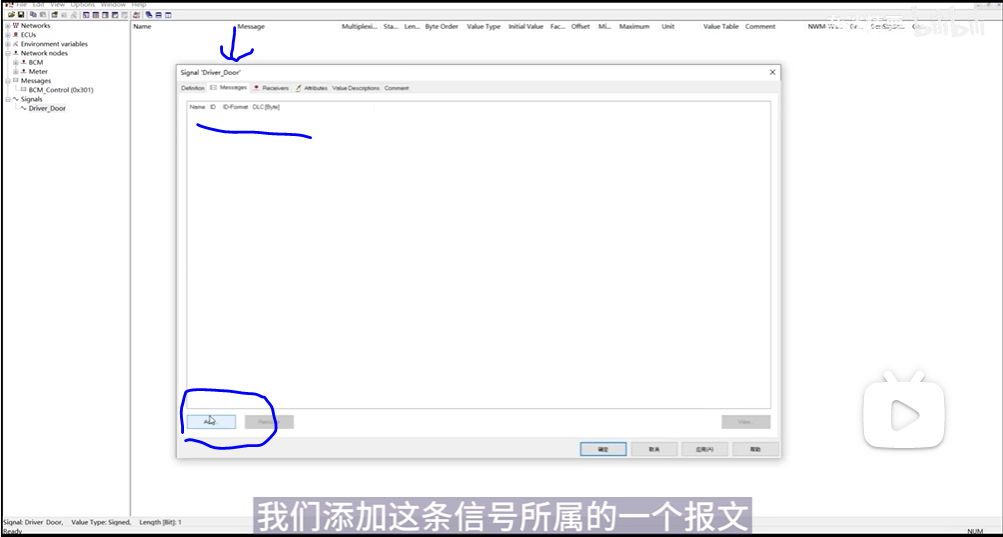
报文添加完毕以后，就添加相应的一个信号。由于此处只有一个例子，所以报文、信号比较少，就只有一个信号名称为Driver door。驾驶侧门的开关状态信号，Data length 是只有一个bit就是1位，然后byte order这里是指数据的排列顺序。数据的排列顺序又2种，一种是intel，另外一种是Motorola

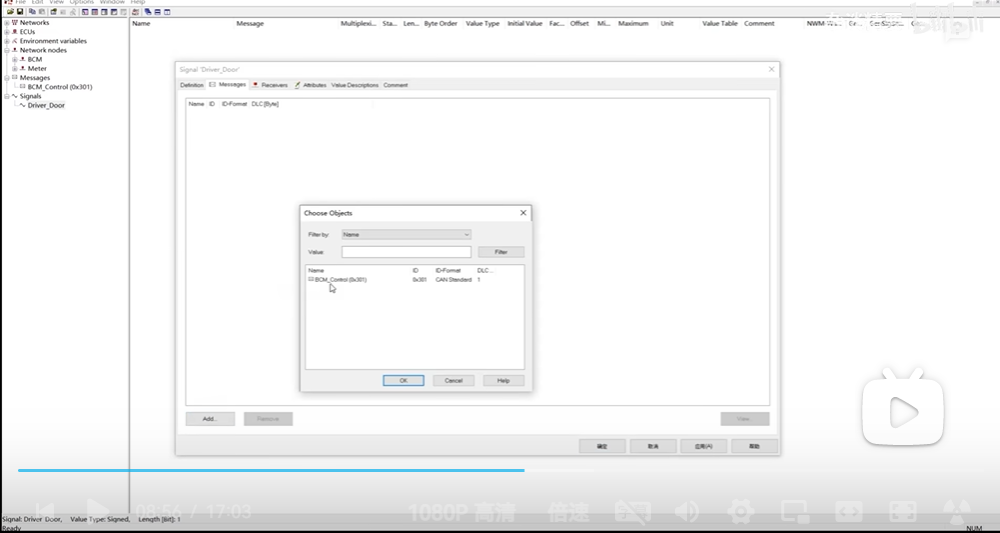


Intel和motorola在字节长度小于8位或者是说不跨字节的时候其实是一样的。但是当数据长度超过8位或者跨字节的时候，两者就又比较大的差异了。这里以一个跨字节的8位数据举例，motorola格式是从上到下，从高到低排列，也就是说，高位是从左到右，从上到下依次进行，是比较符合我们思维习惯的一种排列格式，而英特尔格式却相反，它是高位在下面，然后从高到低，然后跳到上面以后再从高到低。所以两者有一定的区别，这里选择摩托罗拉格式。

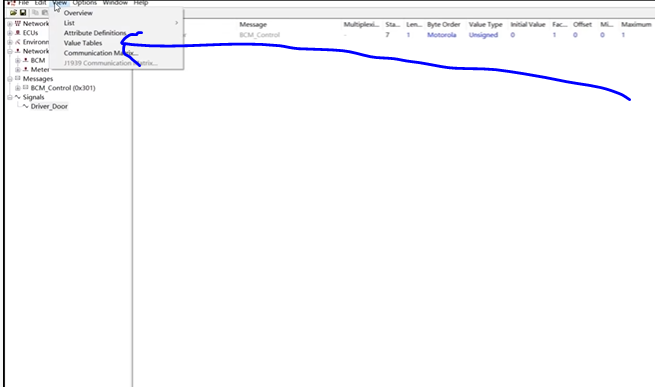


然后value type就是指信号的一个类型，是有符号的还是无符号的，我们这里是状态位，所以选择无符号的。Initial value指初始值，为0。这里factor跟我们刚才矩阵表里面的resolution，是一个意思，都是指一个倍乘系数。那这里状态位系数填1即可，Offset我们这里填0。这样的话就可以计算出最小最大值，这里可以填写也可以计算，点击 Calculate minimum and maximum，可以进行自动计算，那这里最小值就是0，最大值是1 Value Table这个地方对应刚才矩阵中的coding部分。这个coding部分需要在另外的一个界面，填写完以后在这里进行一个选择。

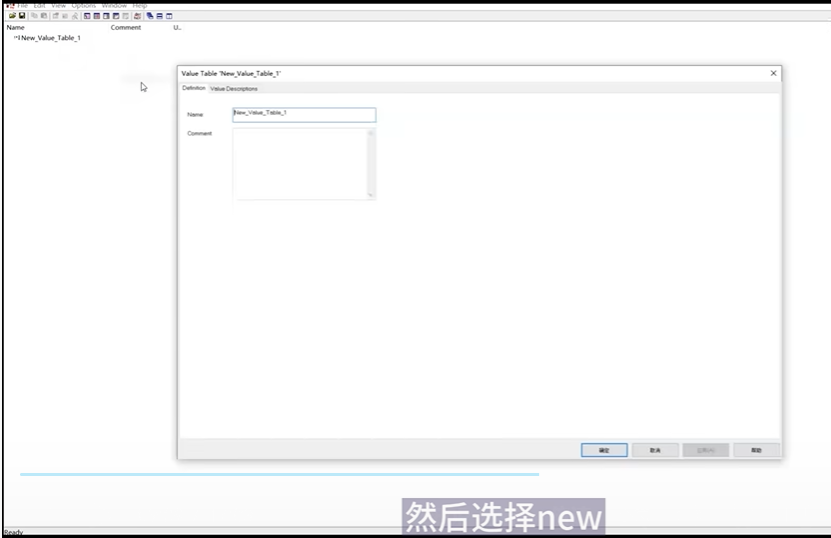




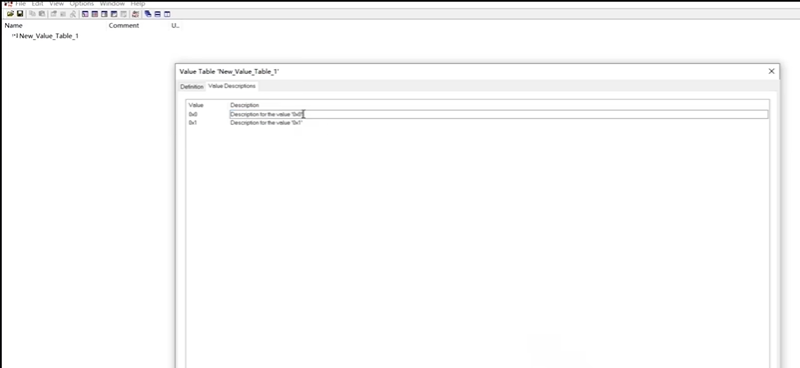
然后点到message这里，我们添加这条信号所属的一个报文，点击确定后，接下来创建一下value table。这里可以添加图形界面的value table，或者是在view这个菜单栏中选择value table。



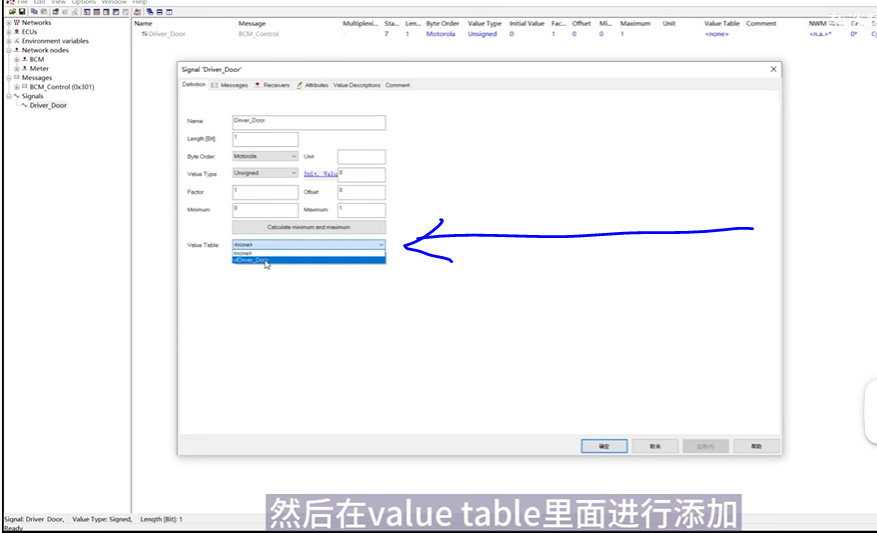
新的界面中右键选择new



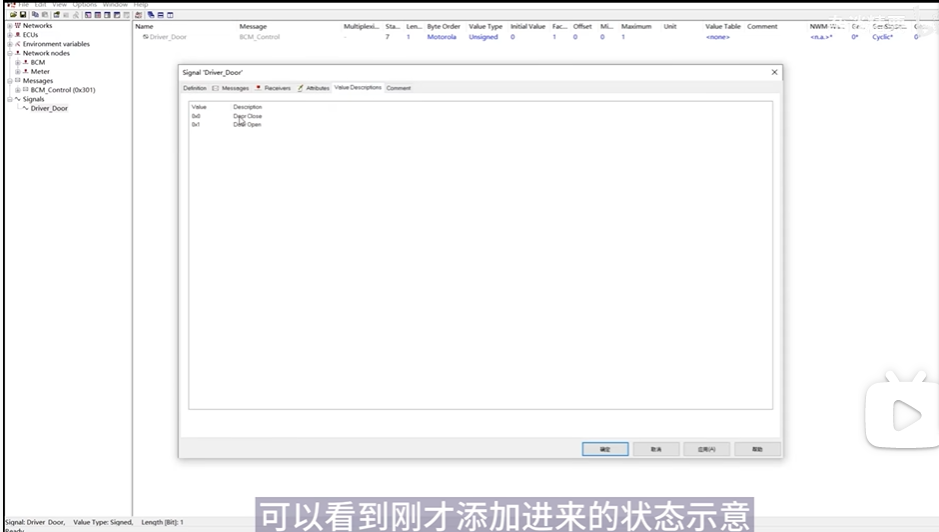
然后命名，一般以vt开头，然后这个是driver door。切换到Value Description，由于只有0、1信号，所以可以添加2个信号，然后进行编辑。



0b代表door close，1b代表door open。点击确定后，回到dbc数据库，双击点开driver door 这个信号，然后在Value table中进行添加，

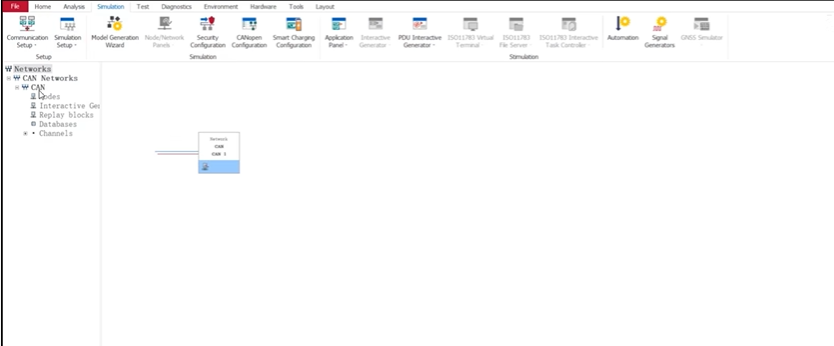


也可以切换到value Description中，可以看到刚才添加进来的状态示意。

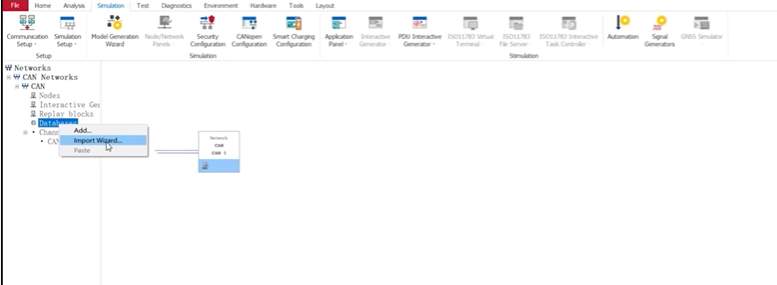


然后点击确定并保存，这个时候简单的数据库就已经做完了。

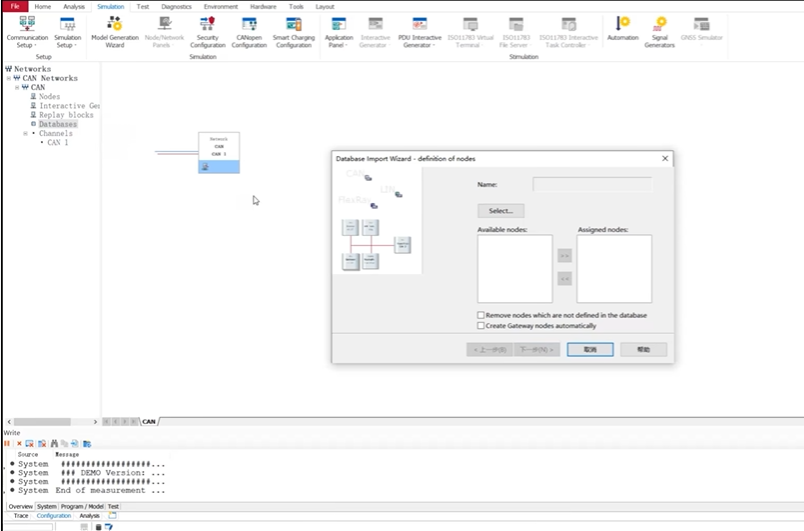
数据库创建完了以后，可以来进行一个仿真，模拟报文的发送，点击simulation然后simulation setup



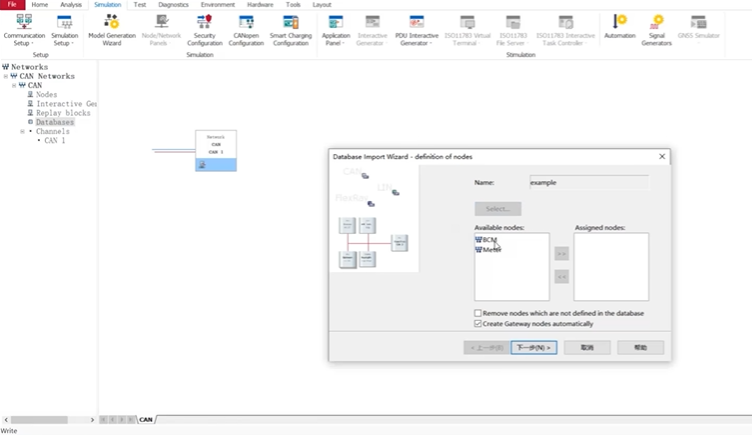
在network下面，这个database添加一个节点和数据库，database这里选择import wizard，



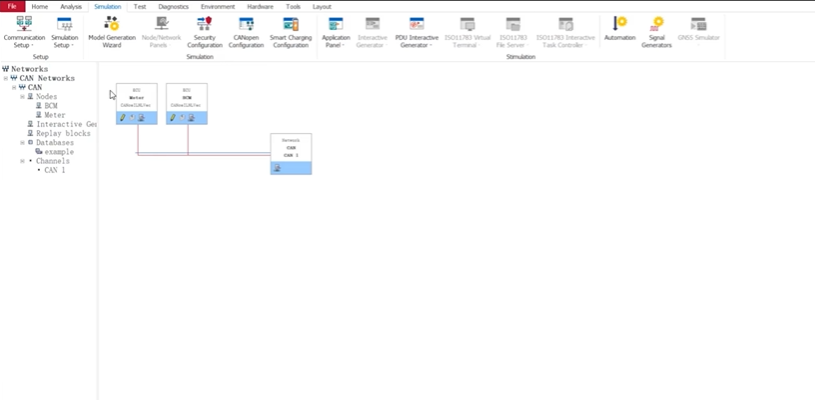
住:此处如果点了add加入数据库的话，只是把这个解释性的文件加进来，并没有把刚才的那个网络还有节点加进来，就不能自动发送报文



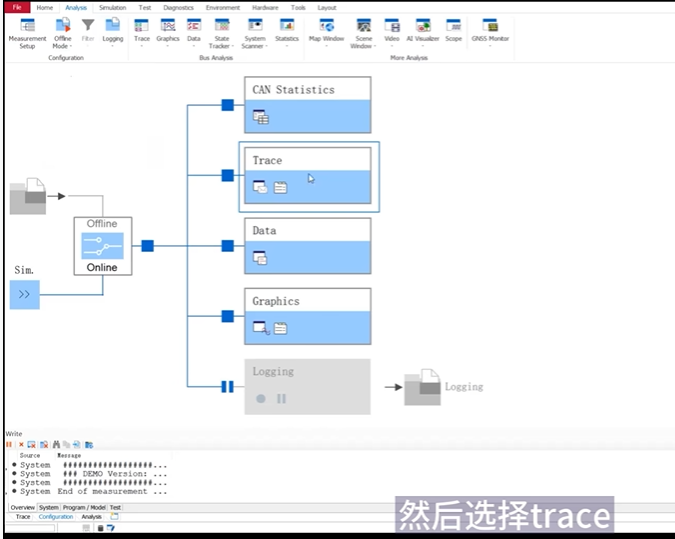
这里选择import wizard.然后在这里Select选择数据库，



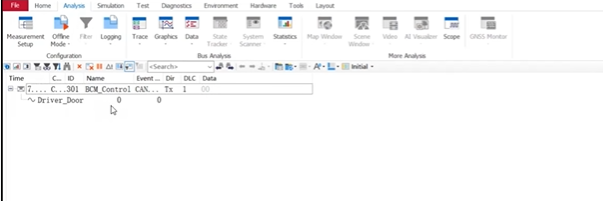
这里可以看到刚才建立的两个节点，一个BCM，一个Meter都进来了，然后选中添加，点击下一步、完成。可以看到两个仿真节点已经添加进来



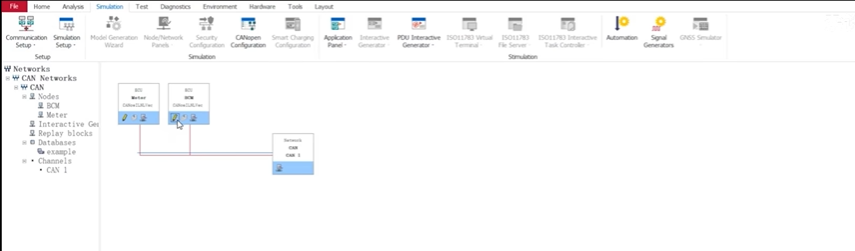
导览栏选择Analysis measurement setup，



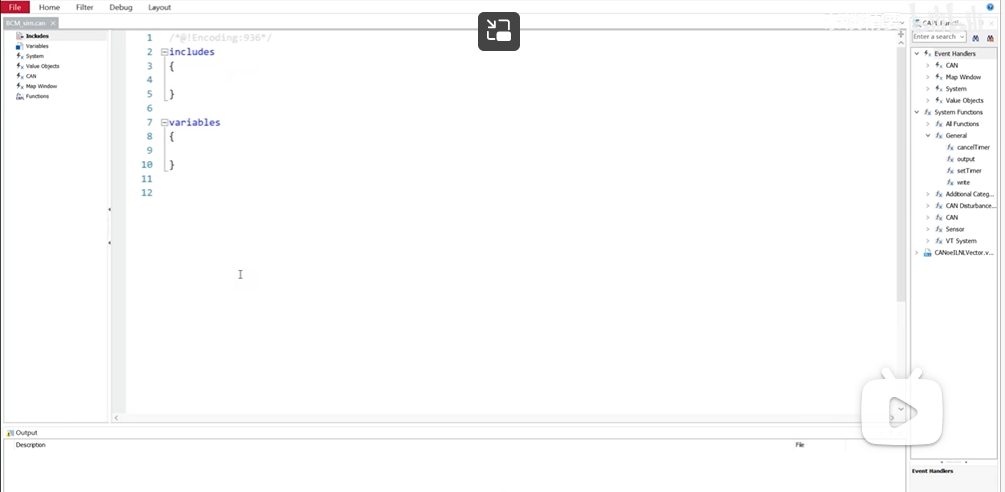
然后选中Trace，并点击下运行，可以看到报文我们可以成功读取到了



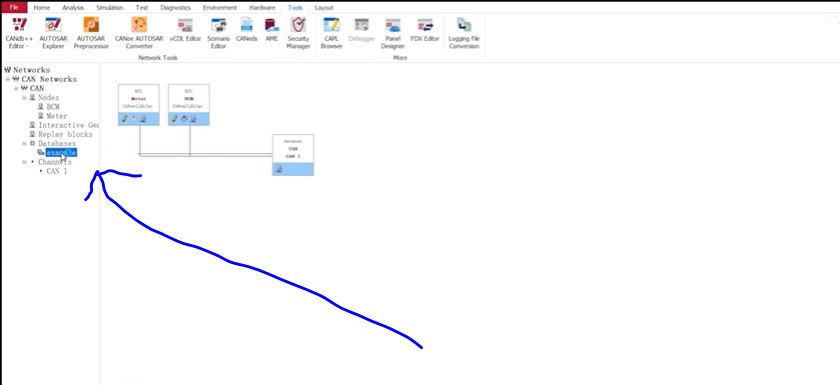
但是这里值一直是固定值，那我们怎么来改变它呢，我们现在回到simulation setup这个仿真节点。做一个简单的仿真程序。

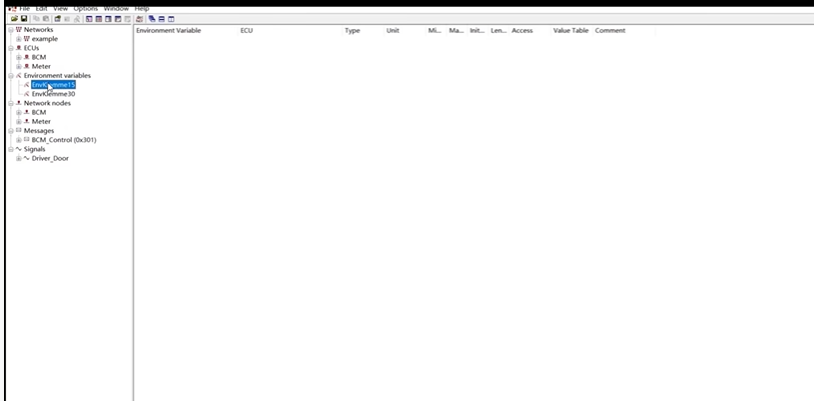


点击BCM，新建一个CAN文件，这时候我们就打开了CAPL编程的一个界面，但是在编写程序之前，我们需要再新建一个环境变量



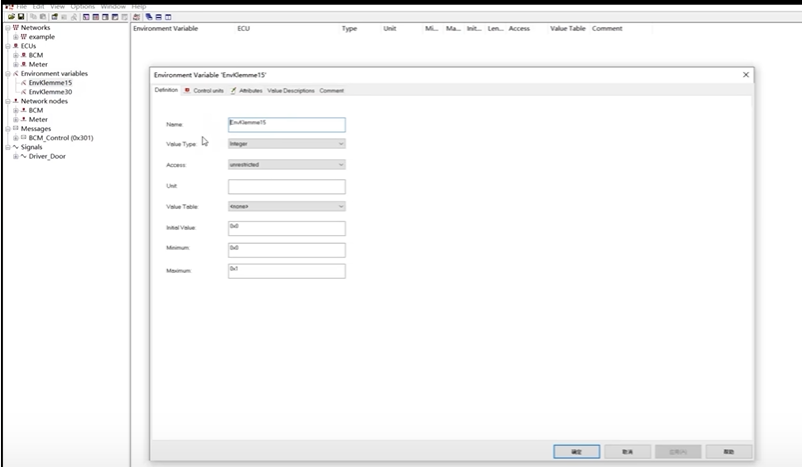
也就是刚才说了，环境变量是连接CAPL程序与面板，还有这个测量界面的一个桥梁，那现在点开Database





点击数据库中的environment variable这里。

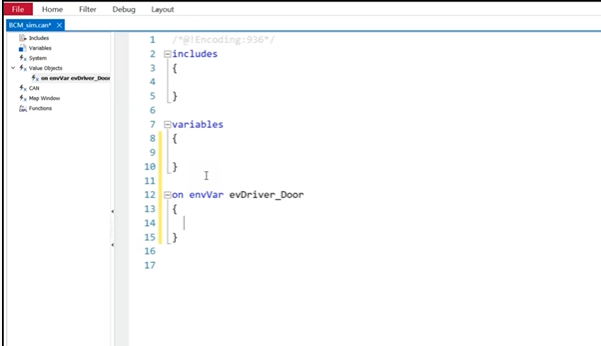
这里可以选择这个数据库默认提供的环境变量进行编辑



环境变量命名为evDriver\_Door,Value Type选择整形Integer，Access选择unrestricted，Value Table选择刚才建立的vDriver\_Door，初值设为0，最小值，最大值设为0和1。

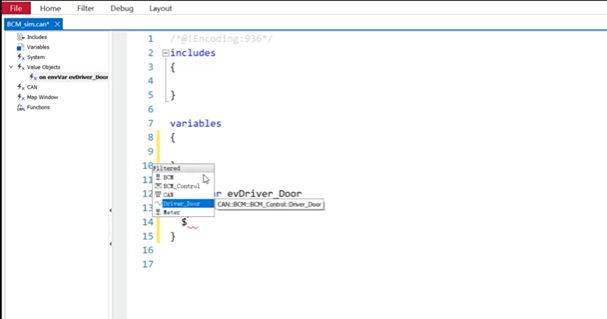
保存后回到刚才编程的页面。

由于CAPL是面向事件型的程序，所以只有当环境变量从0和1发生改变的时候，然后才触发这个事件，

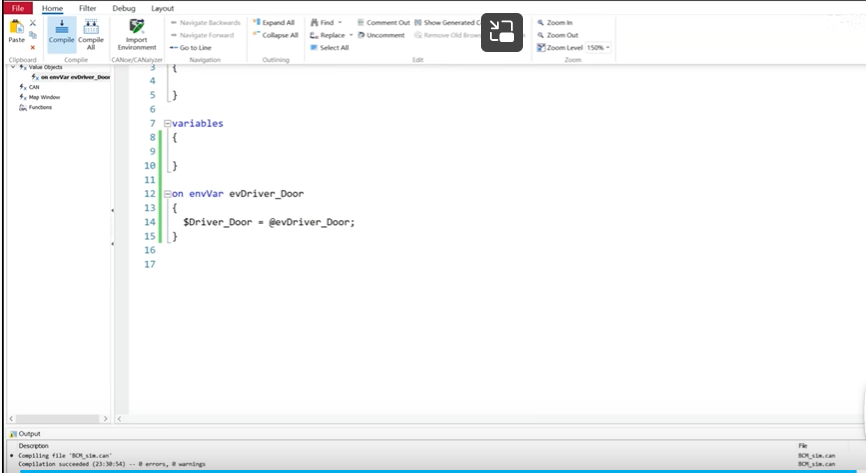


敲上on envVar evDriver\_Door……

也就是这个值发生改变的时候，我们就让环境变量的值赋给刚才报文中的一个信号，那这里报文信号的提取要用到一个符号，$,打完以后会显示信号值

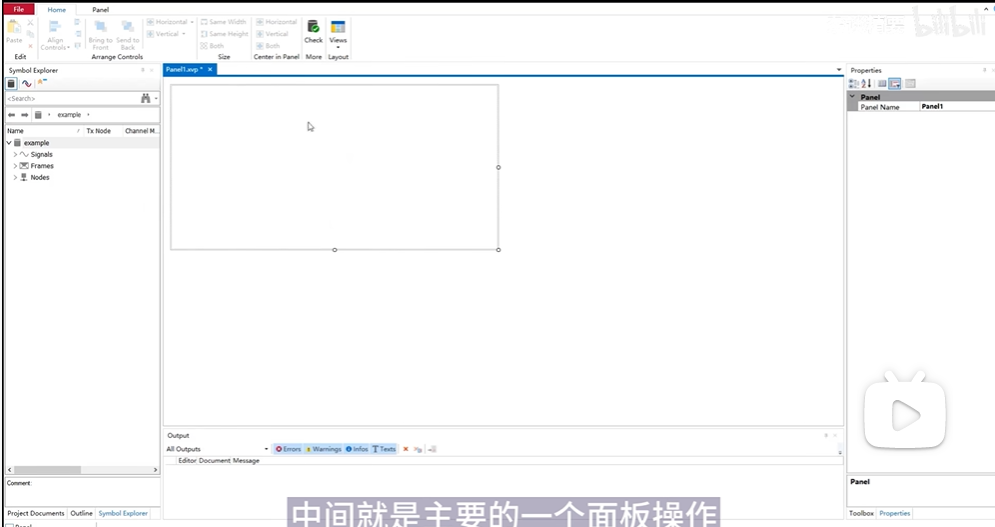


这里选择Driver\_Door，然后环境变量的值需要一个@符号，将环境变量的值提取出来，然后赋给Driver Door这个信号。这里编完程序后即保存。

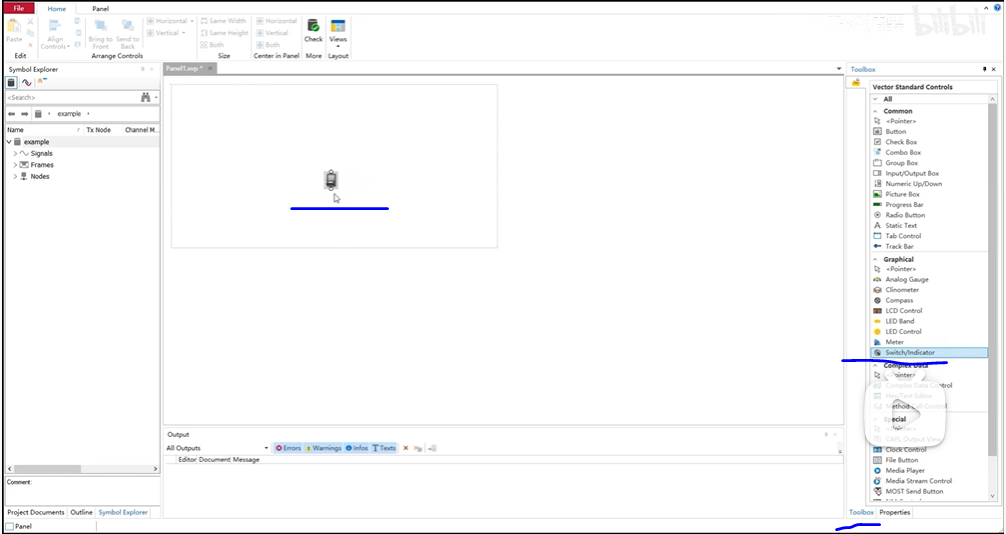


然后点击运行，没有错误。

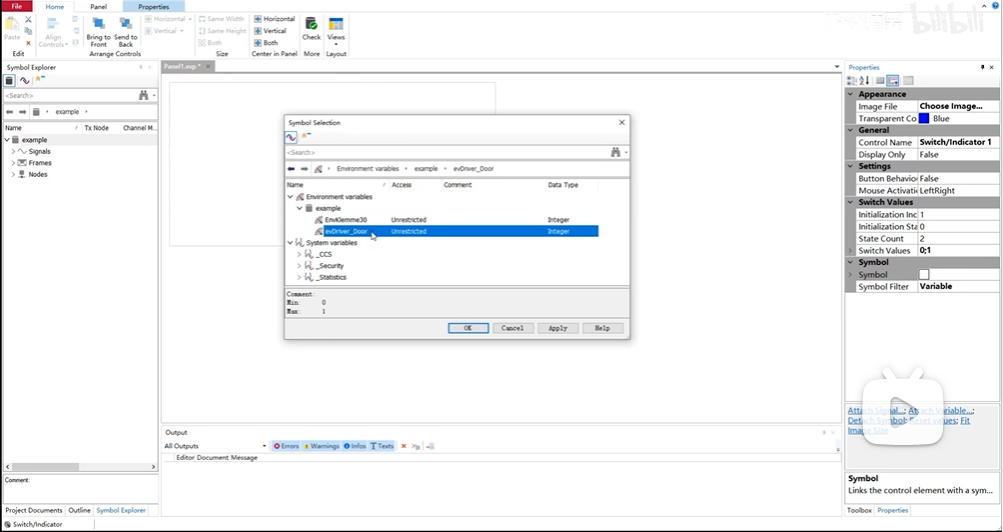
接下来制作一个简单的环境面板将环境变量和面板操作关联起来，在Tools点击panel designer。



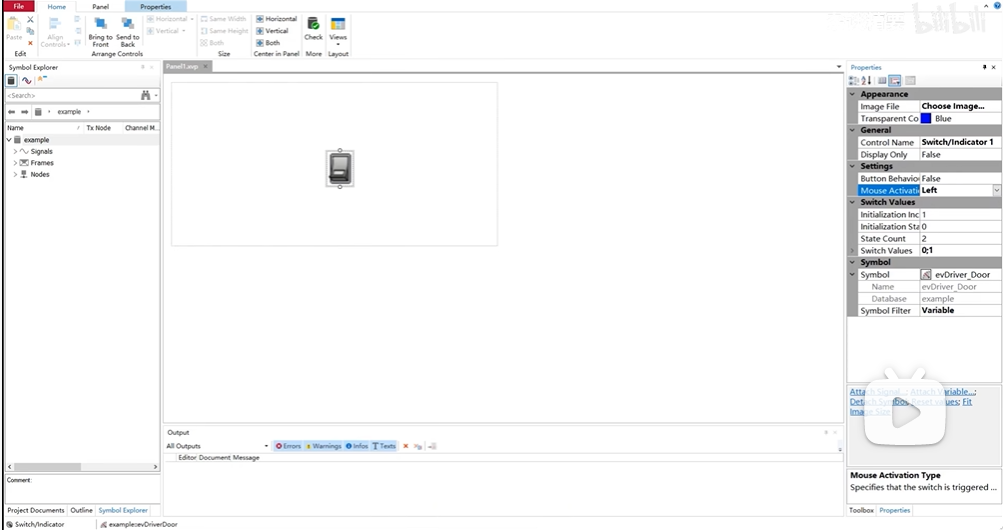
此时已经进入了面板的编辑界面，中间就是面板操作绘制的区域，



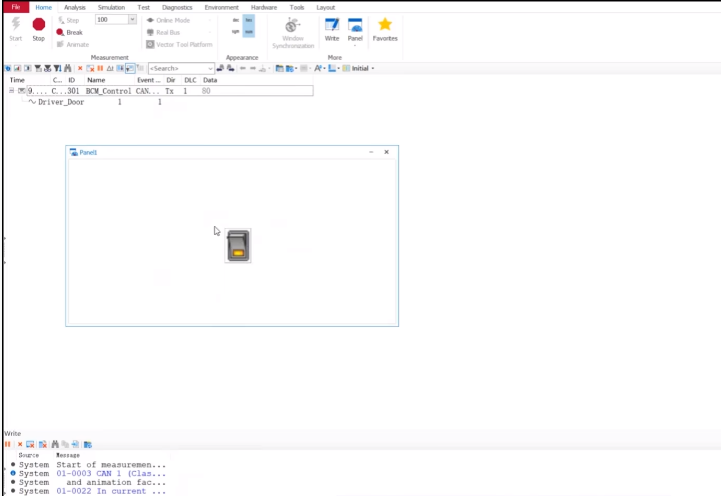
点击右下方的工具栏tool box，用一个按钮来控制刚才信号的跳变，选择switch。然后选中这个开关，点击右下方the properties。



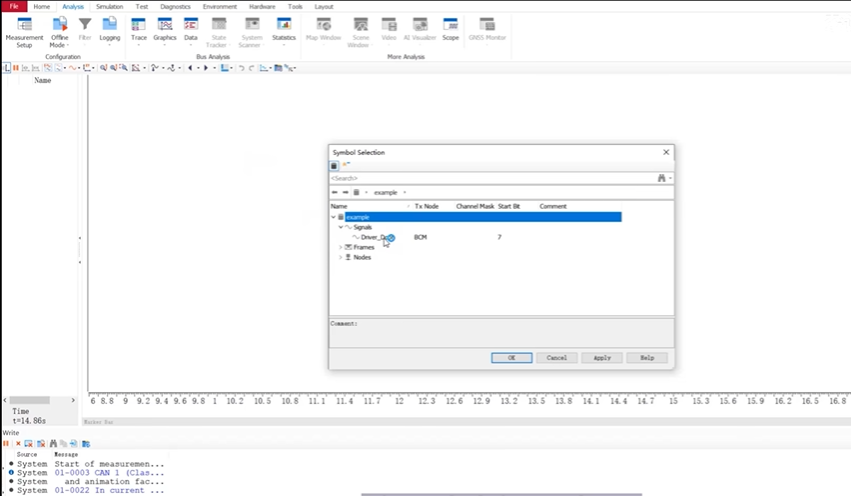
就可以看到相关参数的一个设置，在Symbol Fitter这里选择环境变量，选择Variables，然后选择evDriver\_Door，设置完symbol之后，将鼠标的操作setting设置一下，button behavior就是指单击完以后，它会不会恢复，这里默认为false就可以了。Mouse Active是指左右键操作还是单击操作，这里选择单击操作left，设置完毕后保存。



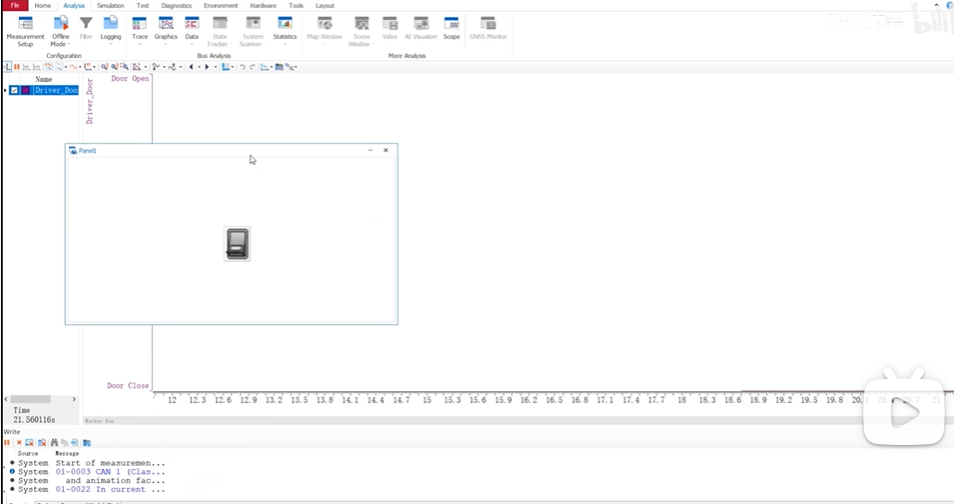
回到仿真界面后，这里点击Panel调出我们已经建好的这个面板，点击Start开始运行，可以看到，点击开关，信号就发生了一个跳变。



更直观的话可以看图形界面，点击Graphics，右键add signals添加信号 driver door，



然后将面板调出来，



这里先按鼠标滚轮，将信号缩小一点，然后点击按钮，可以看到这个时候信号已经发生跳变了。

