

CANoe快速入门

CANoe概述

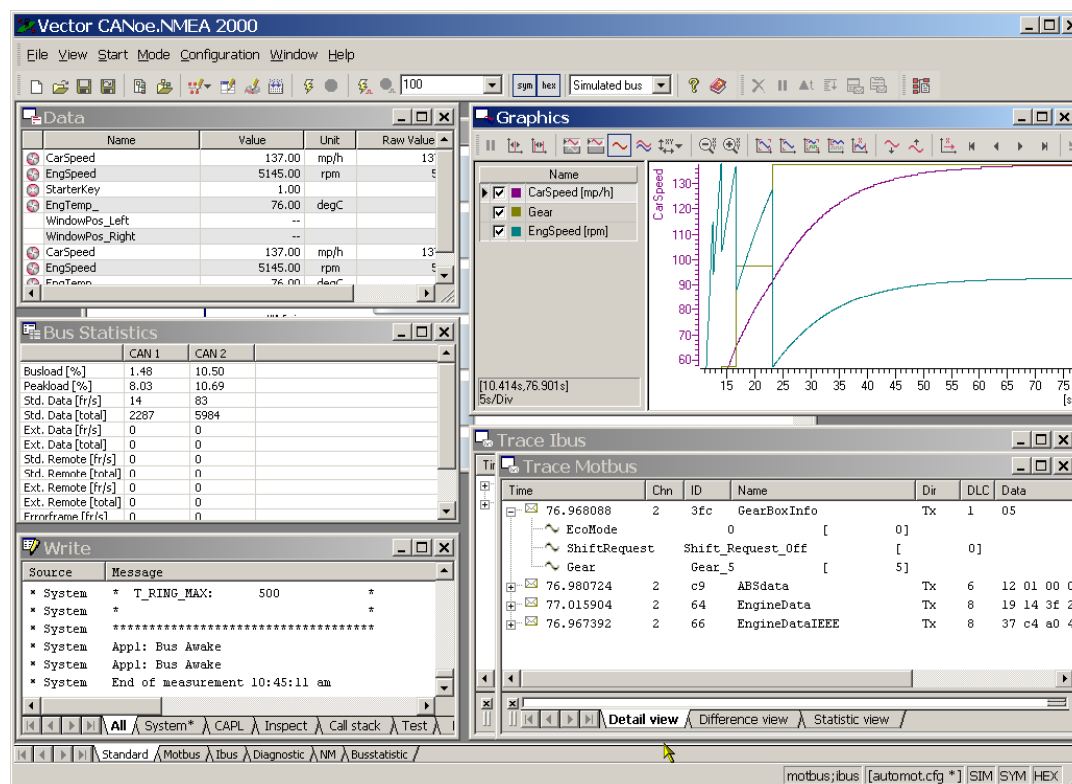
□ CAN总线开发工具

□ 测试

□ 分析

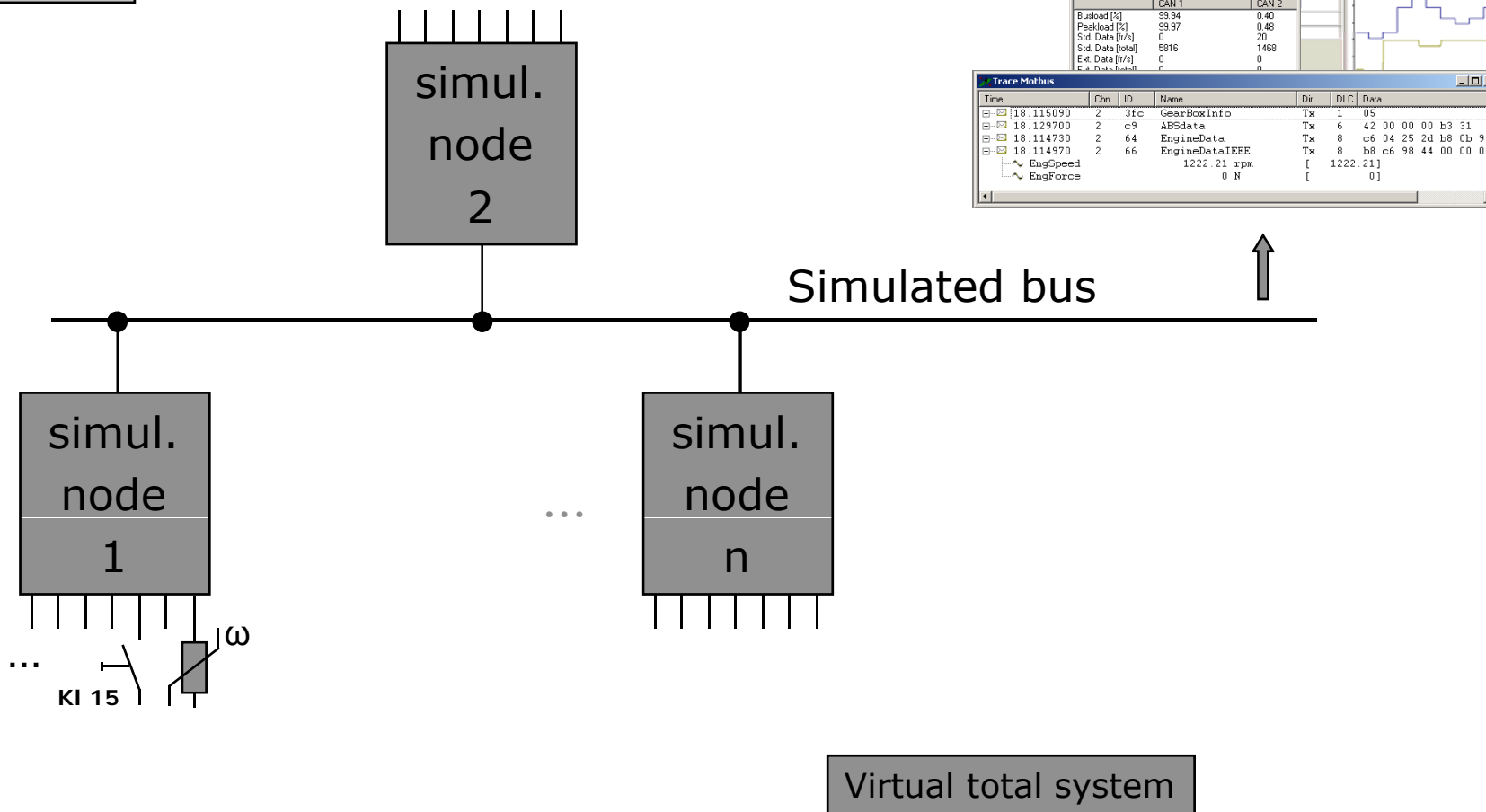
□ 仿真

□ 记录

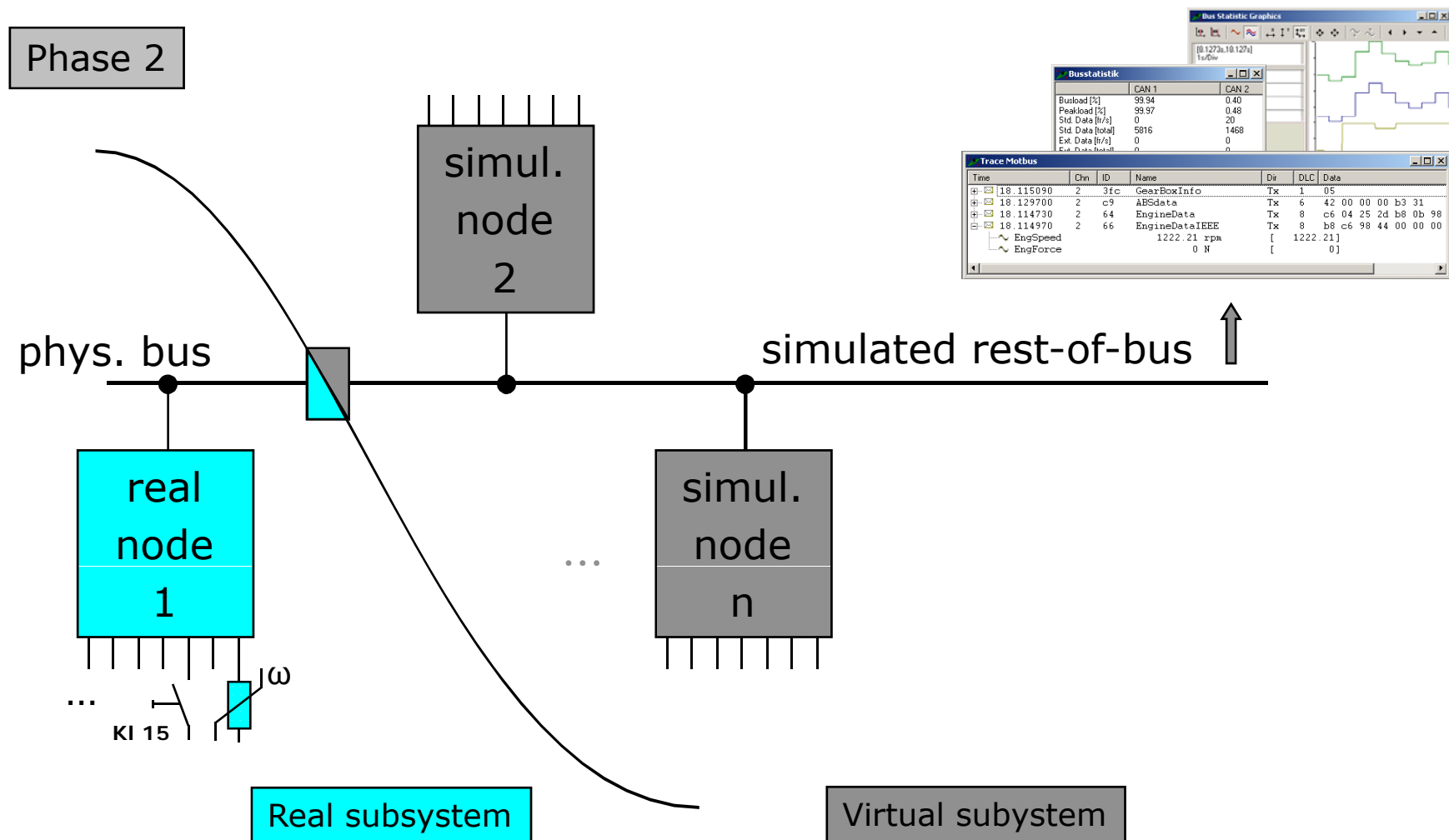


CANoe在总线开发中的作用 (1)

Phase 1

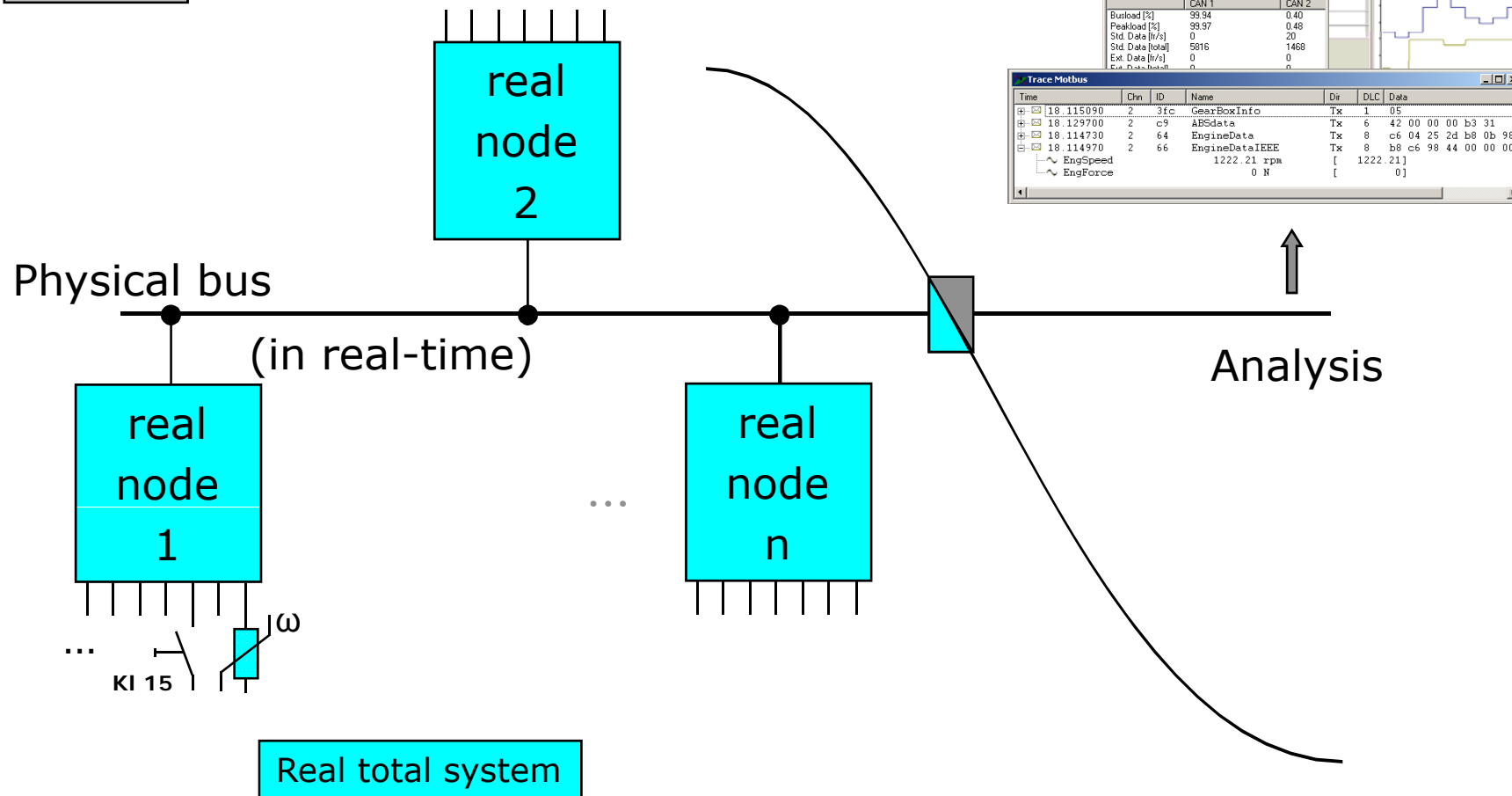


CANoe在总线开发中的作用 (2)



CANoe在总线开发中的作用 (3)

Phase 3



CAN卡

❑ 硬件接口卡& “狗”

❑ CANcardXL

❑ CANcaseXL

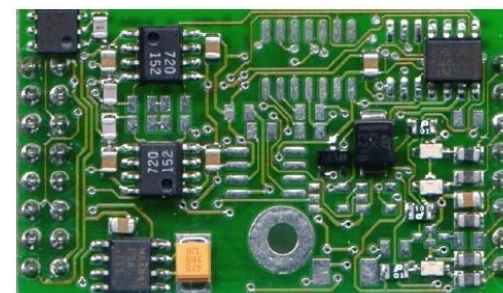
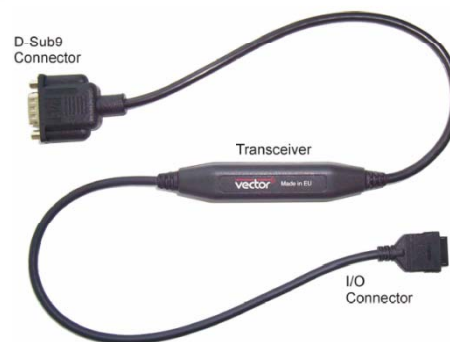
❑ 收发器

❑ CANcab (CANpiggy)

❑ 251, 1041, 1054...

❑ LINcab (LINpiggy)

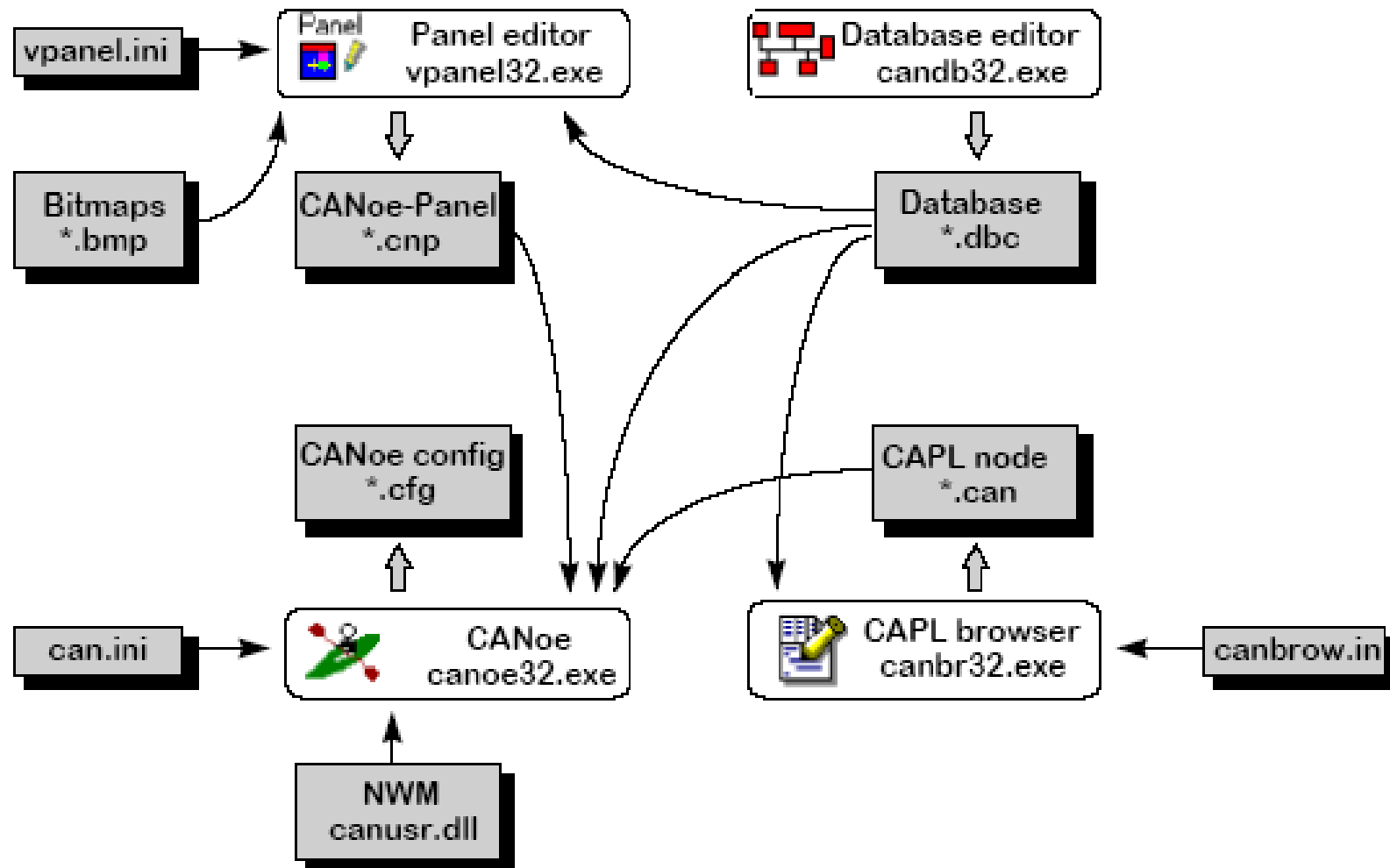
❑ 7259



CANoe组成

- 功能强大、操作简单
 - CANoe
- 数据库支持
 - CANdb++ Editor
- 可编程
 - CAPL
- 虚拟仪表
 - Panel Editor & Panel Designer

CANoe工程环境



多总线

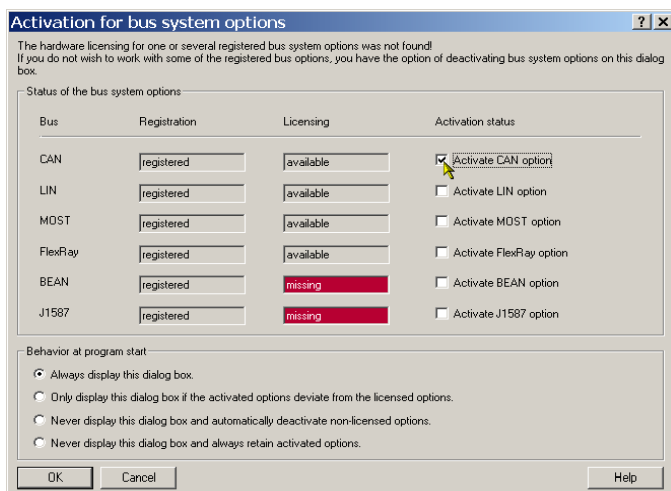
□ 软件

□ 硬件

□ 控制面板

□ Vector Hardware

□ License->Overview



Details

CANoe

CANoe Application
CANoe.CANopen / ProCANopen
Application DENoe PRO
CAN for DENoe PRO
MOST for DENoe PRO
FLEXRAY for DENoe PRO
LIN for DENoe PRO

CANape

CANape Graph

CANalyzer

CANalyzer Application
Application DENalyzer PRO
CAN for DENalyzer PRO
MOST for DENalyzer PRO
FLEXRAY for DENalyzer PRO
LIN for DENalyzer PRO

设置 (1)

□ 硬件

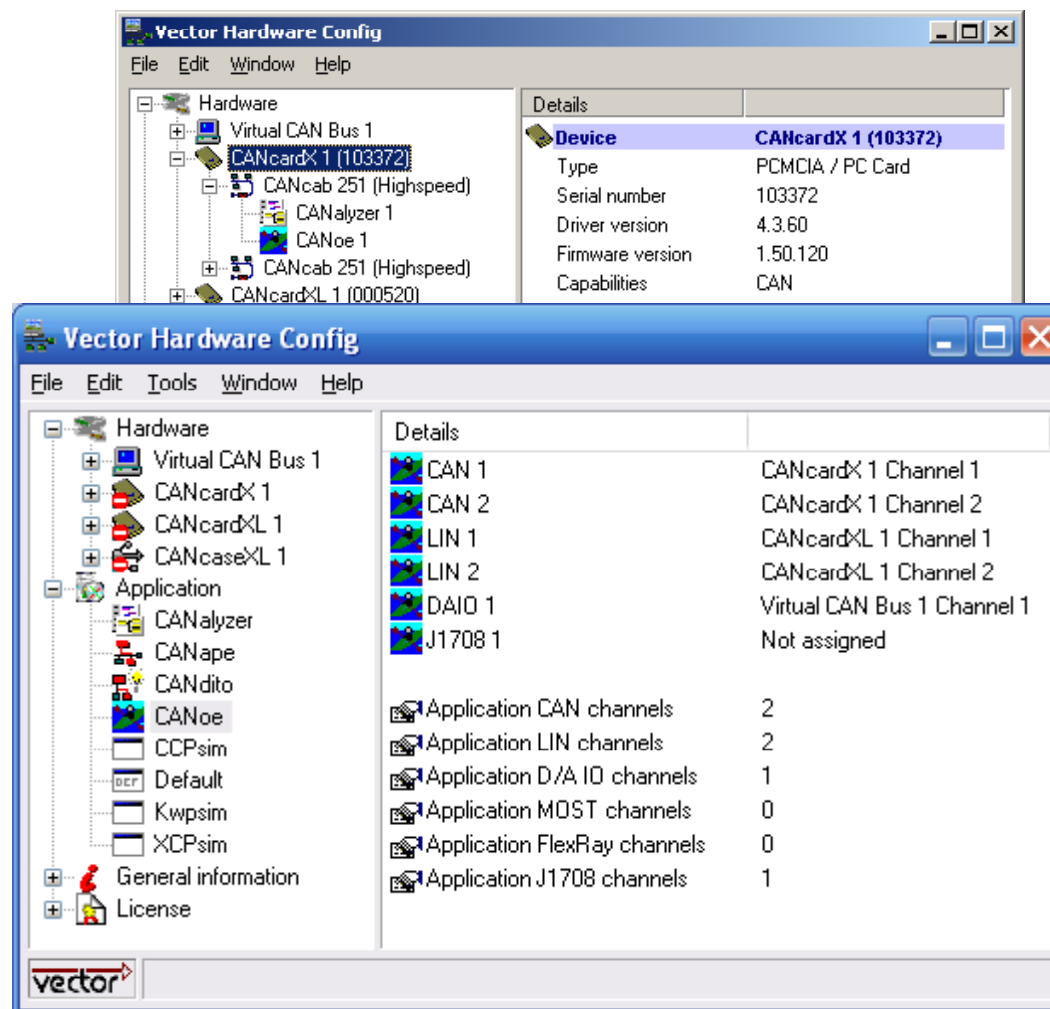
□ CAN卡类型 (编号)

□ 收发器类型

□ 应用程序通道

□ 应用程序

□ License信息



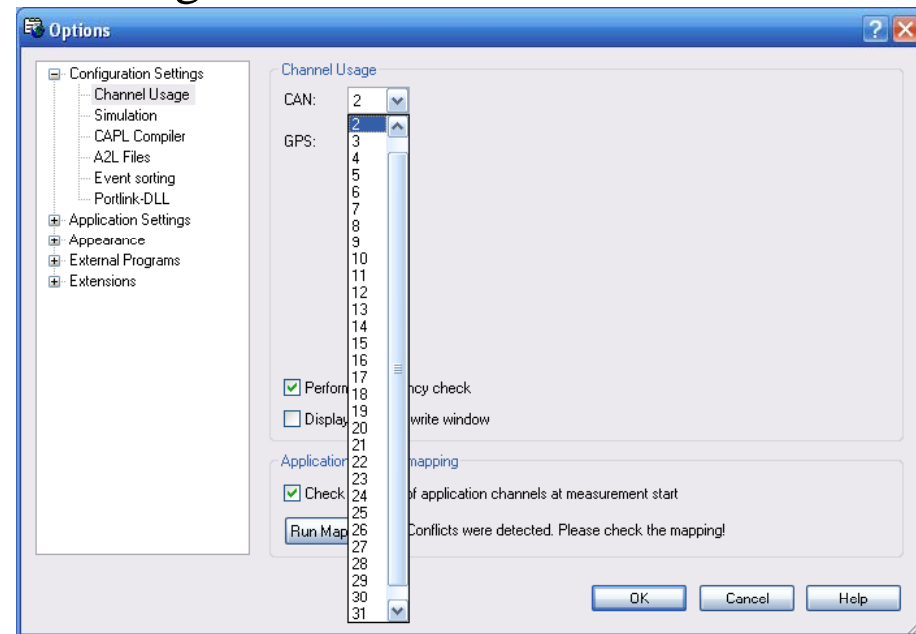
设置 (2)

□ CANoe

□ 通道设置

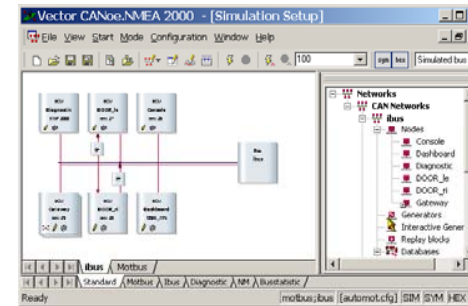
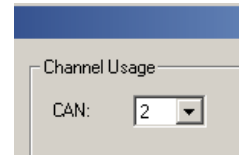
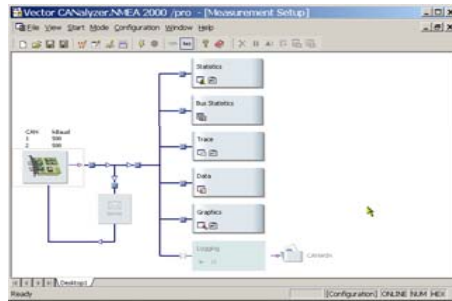
□ Configuration->Options

□ Configuration Settings->Channel Usage



灵活=复杂

Application



App channels

CANalyzer1 CANalyzer2... CANoe1 CANoe2 CANoe1 for LIN...

HW channels

Channel1 Channel2 Channel1 Channel2 PiggyBack1 PiggyBack2

Hardware

CANcardXL #1



CANcardXL #2



CANcaseXL

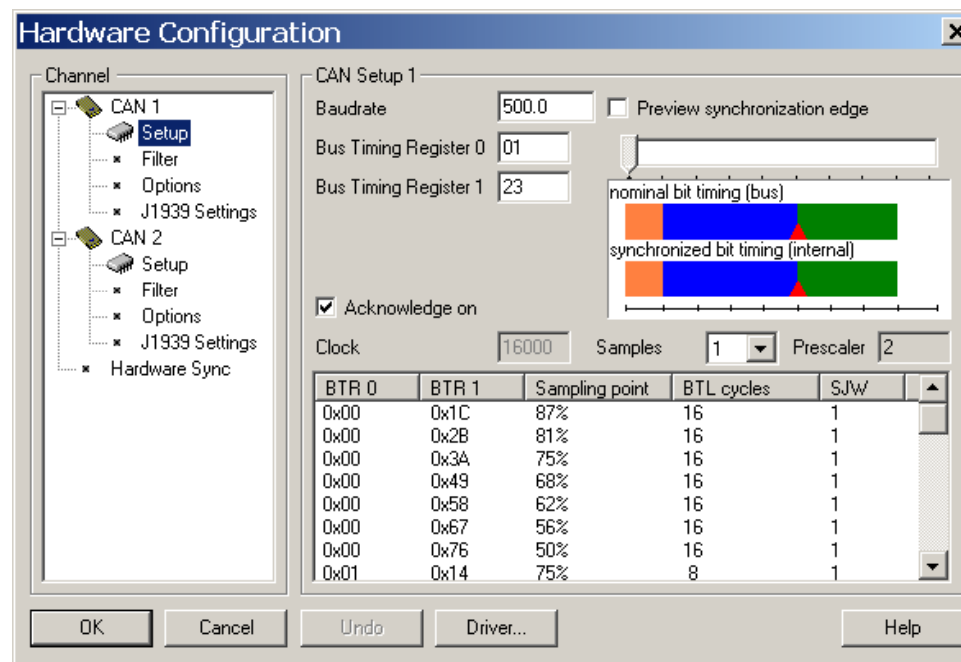


从复杂到简单

□ CANoe

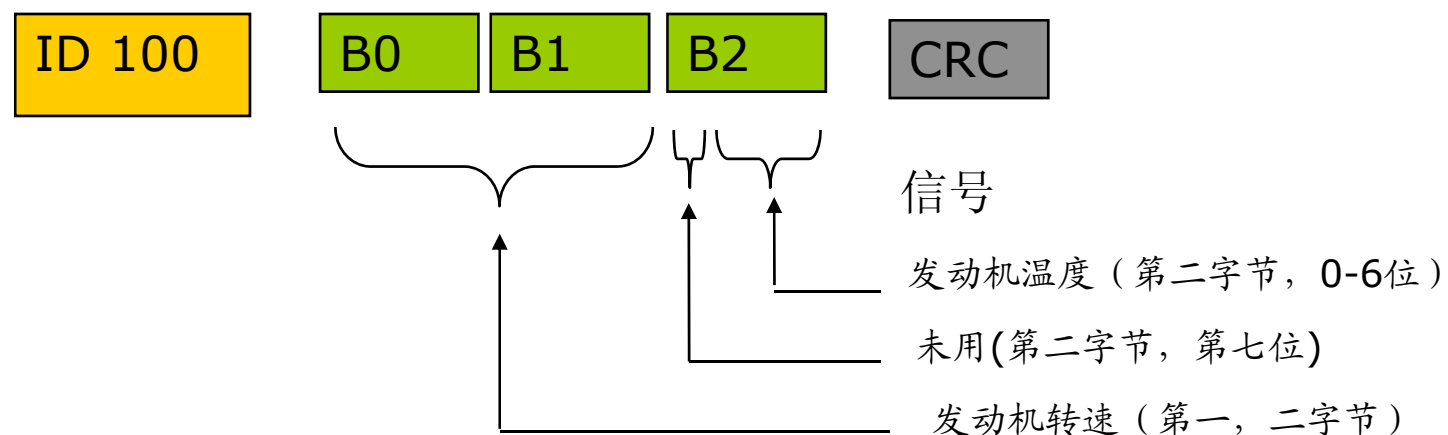
□ 波特率设置

□ Configuration->Hardware Configuration



基本术语——报文（消息）与信号

报文：engine data (ID 100)



转换规则

发动机转速 : $\text{rpm} = 1 * \text{Bit value}$ (0xFF 代表错误)

发动机温度: $^{\circ}\text{C} = 2 * \text{Bit value} - 50$ (0x7F 代表错误)

基本术语——环境变量与系统变量

□ 环境变量

- 节点的I/O信号
- 可用于面板或真实I/O

□ 系统变量

- 节点内部参数
- 或需要观测的某个数值
 - 例如：系统变量1 = 报文1.信号1 - 报文2.信号2

欢迎进入CANoe的世界

- CANoe
- CANdb++ Editor
- CAPL
- Panel Editor & Panel Designer

欢迎进入CANoe的世界

□ CANoe

□ 8大窗口

□ Trace Window

□ Bus Statistics Window

□ Statistics Window

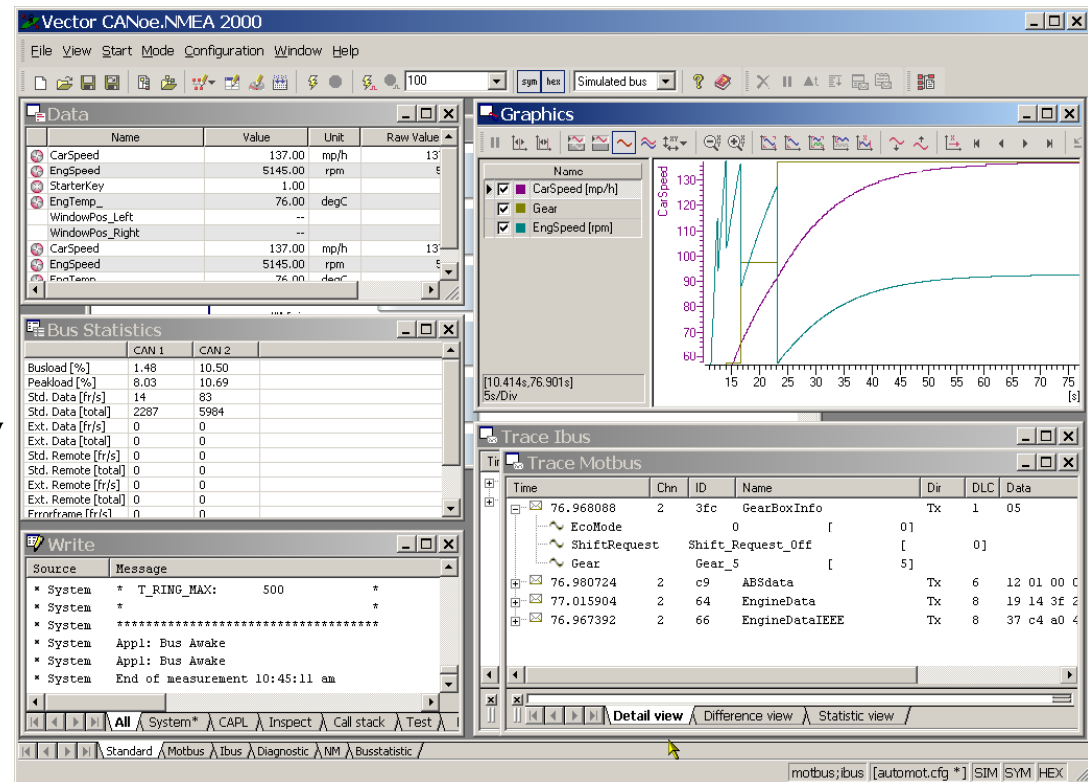
□ Data Window

□ Graphic Window

□ Write Window

□ Simulation Setup

□ Measurement Setup



CANoe窗口介绍（1）

□ Trace Window

□ 报文ID和报文名称（数据库）

□ 信号（数据库）

□ 时间（相对值或绝对值）

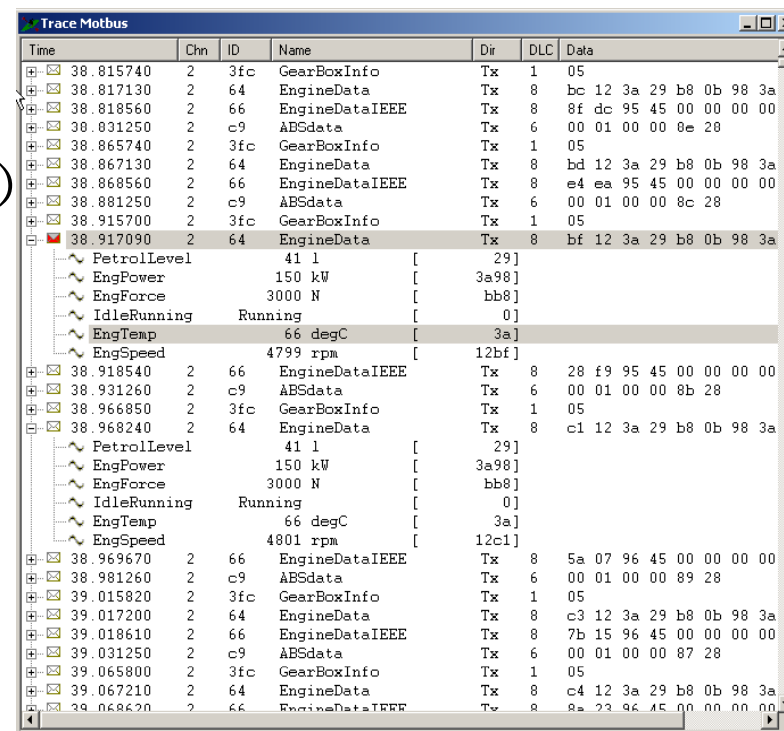
□ 通道

□ DLC

□ Dir（Tx或Rx）

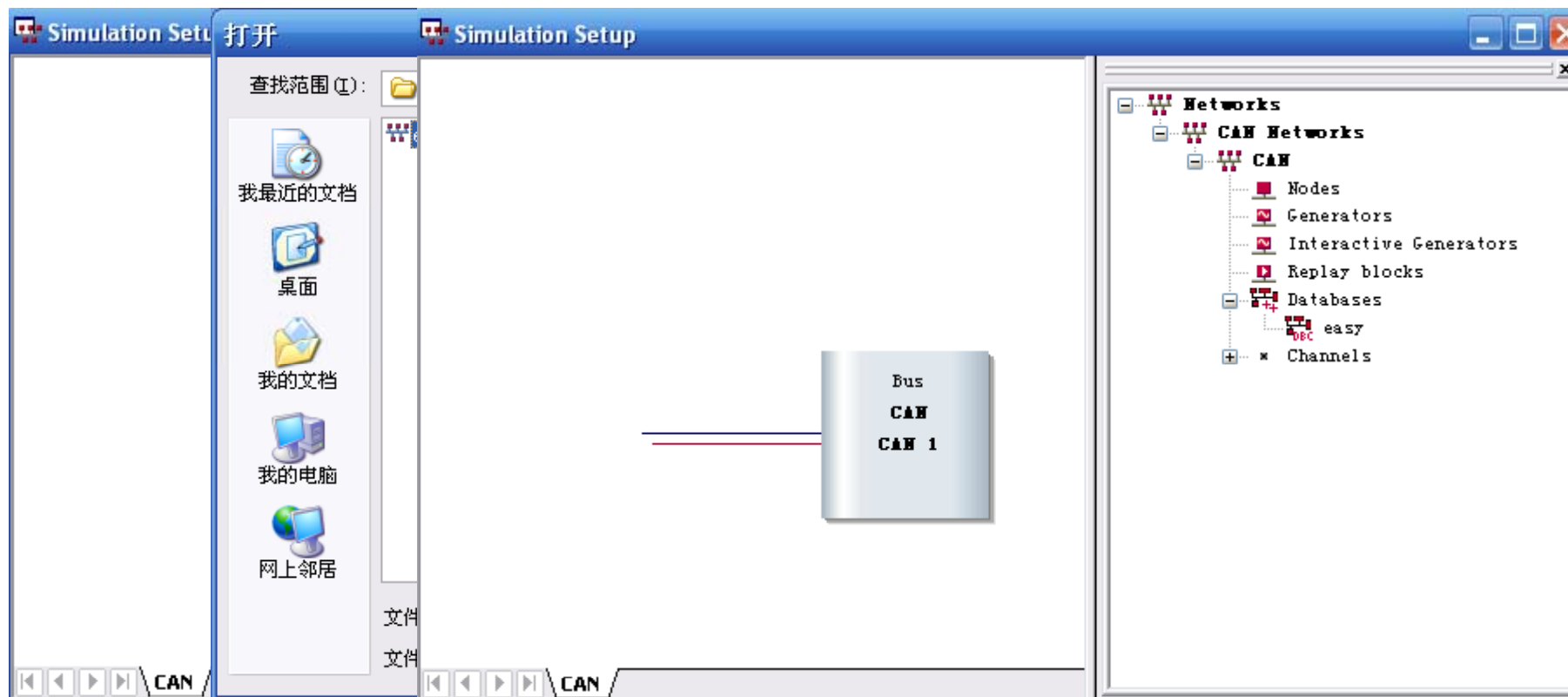
□ 更多内容见

□ 右键点击窗口空白处->Configuration->Columns



蒙太奇 (1)

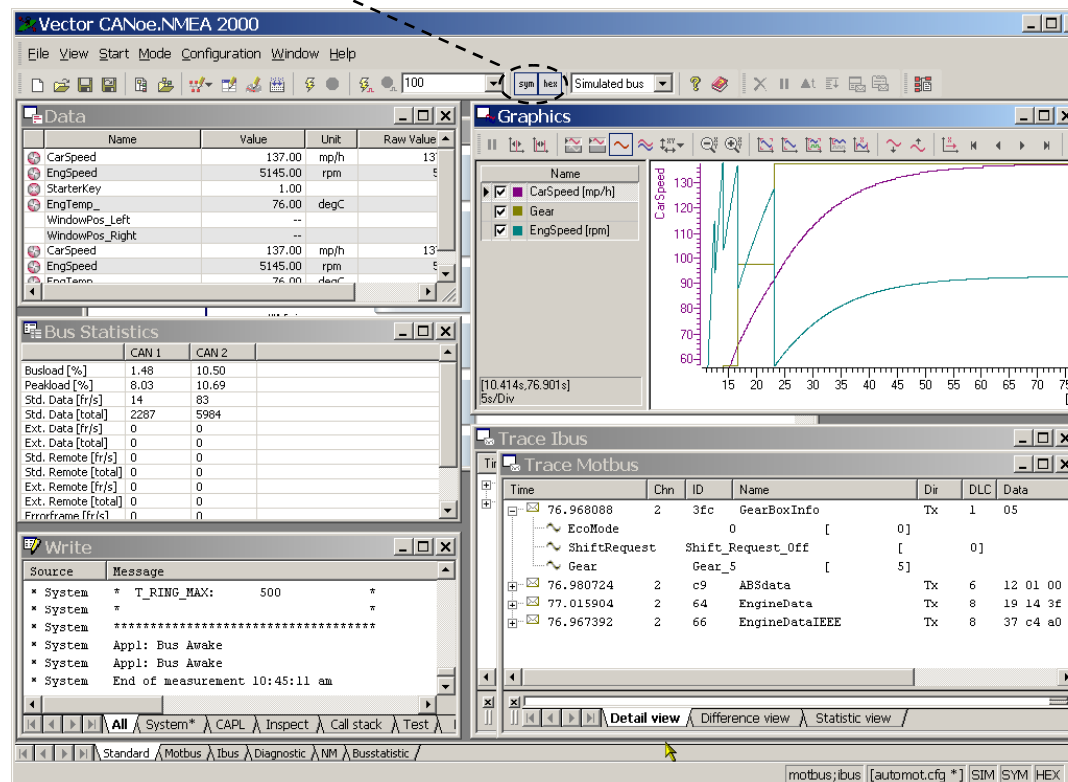
- 在CANoe中添加数据库
- View->Simulation Setup



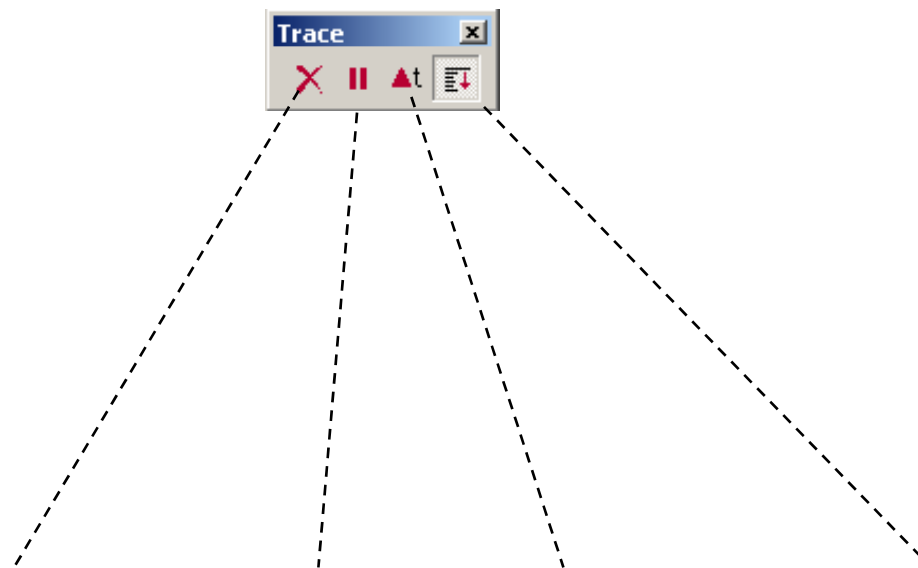
蒙太奇 (2)

▣ 符号化显示与十进制/十六进制切换

Global switches: Hex/Dec and Numeric/Symbolic toggles



CANoe窗口介绍（1）

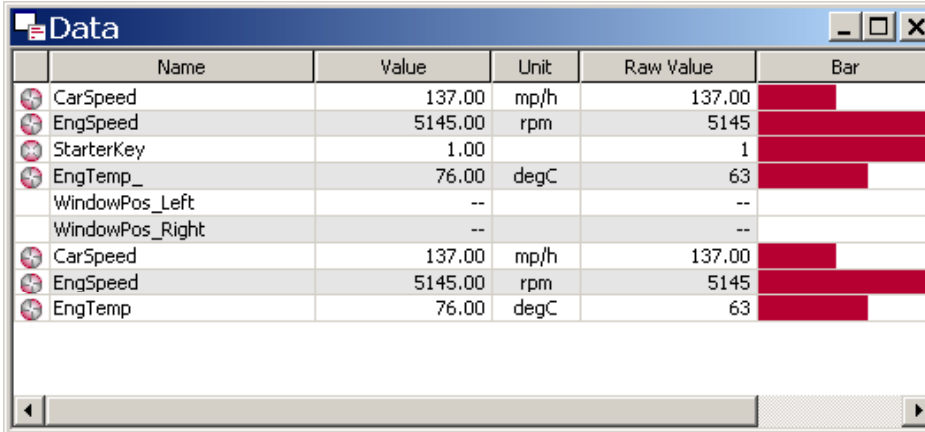


清空Trace窗口 暂停Trace窗口 时间显示切换 报文显示切换

CANoe窗口介绍（2）

□ Data Window

- 数据库！
- 信号名称
- 信号值（Value）
- 信号单位
- 原始值（Raw Value）
- Bar图



	Name	Value	Unit	Raw Value	Bar
	CarSpeed	137.00	mp/h	137.00	
	EngSpeed	5145.00	rpm	5145	
	StarterKey	1.00		1	
	EngTemp_	76.00	degC	63	
	WindowPos_Left	--		--	
	WindowPos_Right	--		--	
	CarSpeed	137.00	mp/h	137.00	
	EngSpeed	5145.00	rpm	5145	
	EngTemp	76.00	degC	63	

CANoe窗口介绍（2）

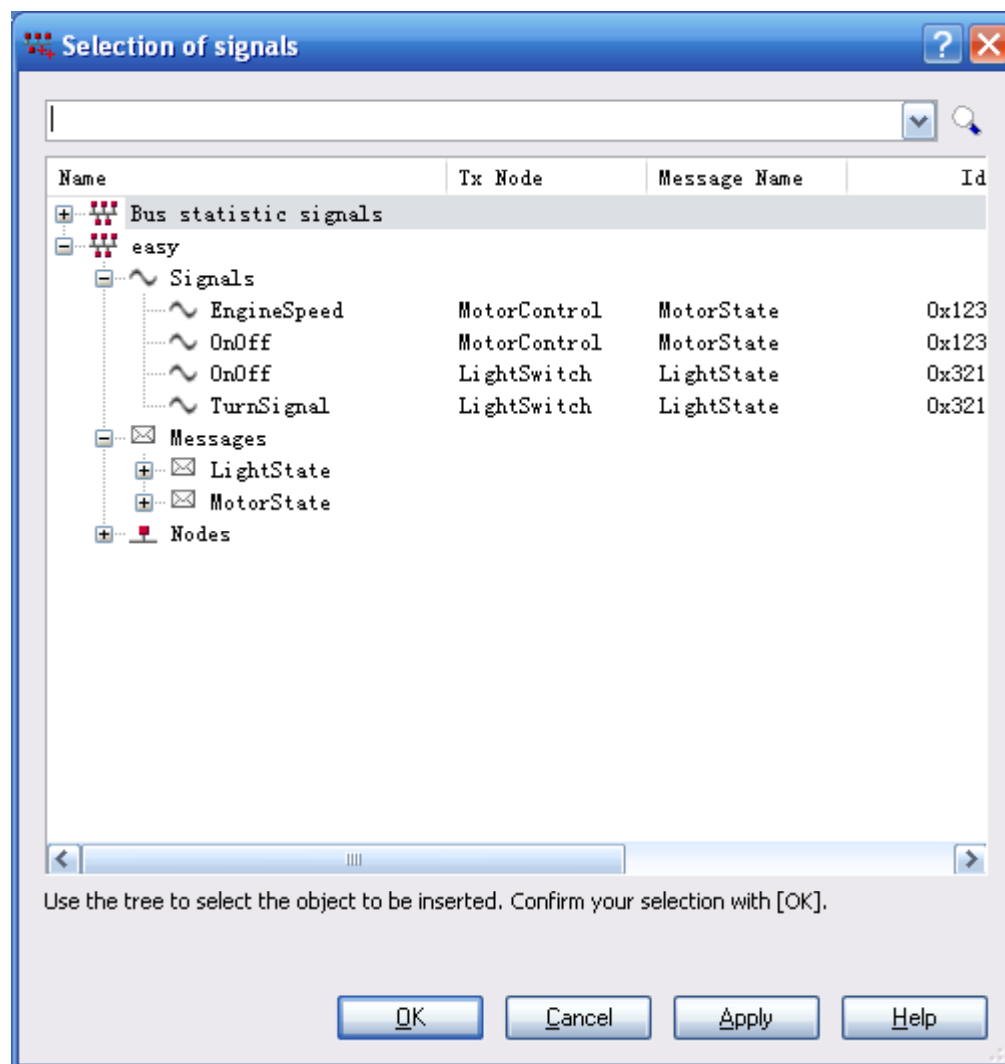
□ Data Window

□ 添加信号

□ 右键单击空白处

□ Add Signals

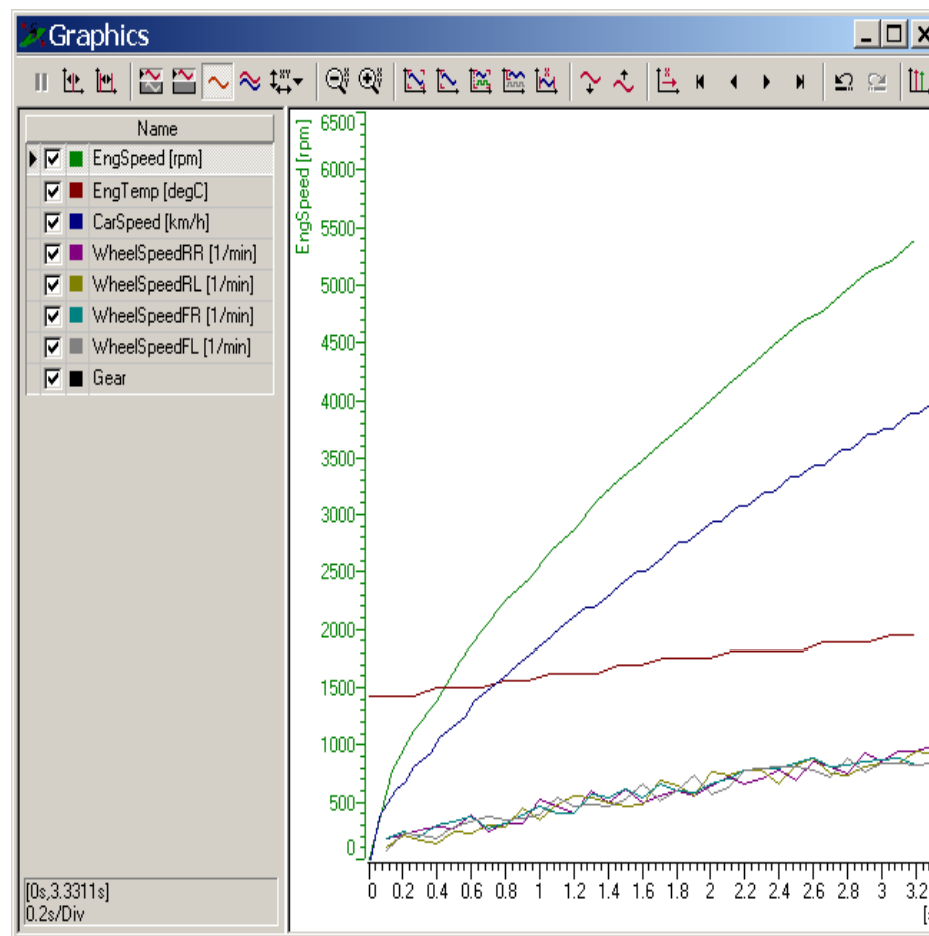
□ 选择需要的信号



CANoe窗口介绍（3）

□ Graphics Windows

- 数据库！
- 显示信号曲线
- 不同的颜色和线形
 - 右键单击空白处
 - 选择Configuration
- 放大、缩小、平移...



CANoe窗口介绍（3）

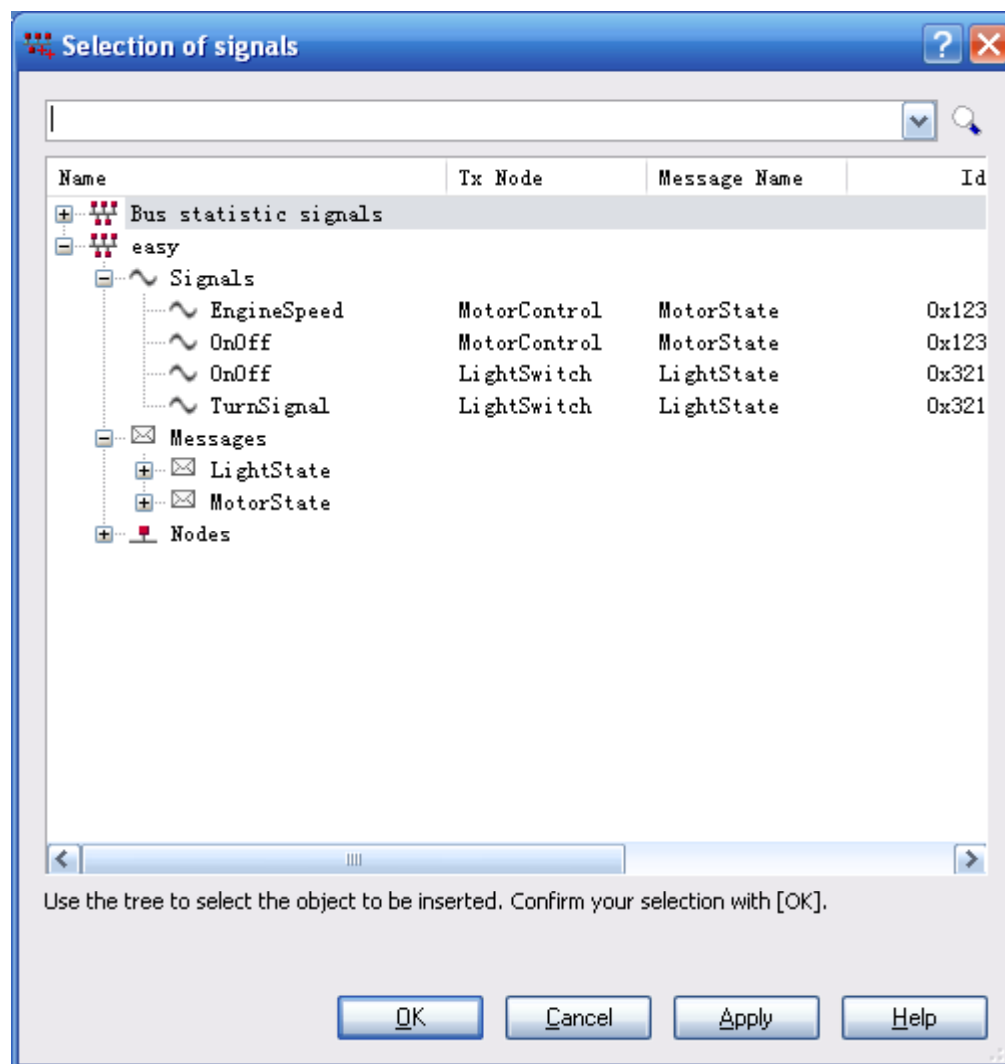
□ Graphics Windows

□ 添加信号

□ 右键单击空白处

□ Add Signals

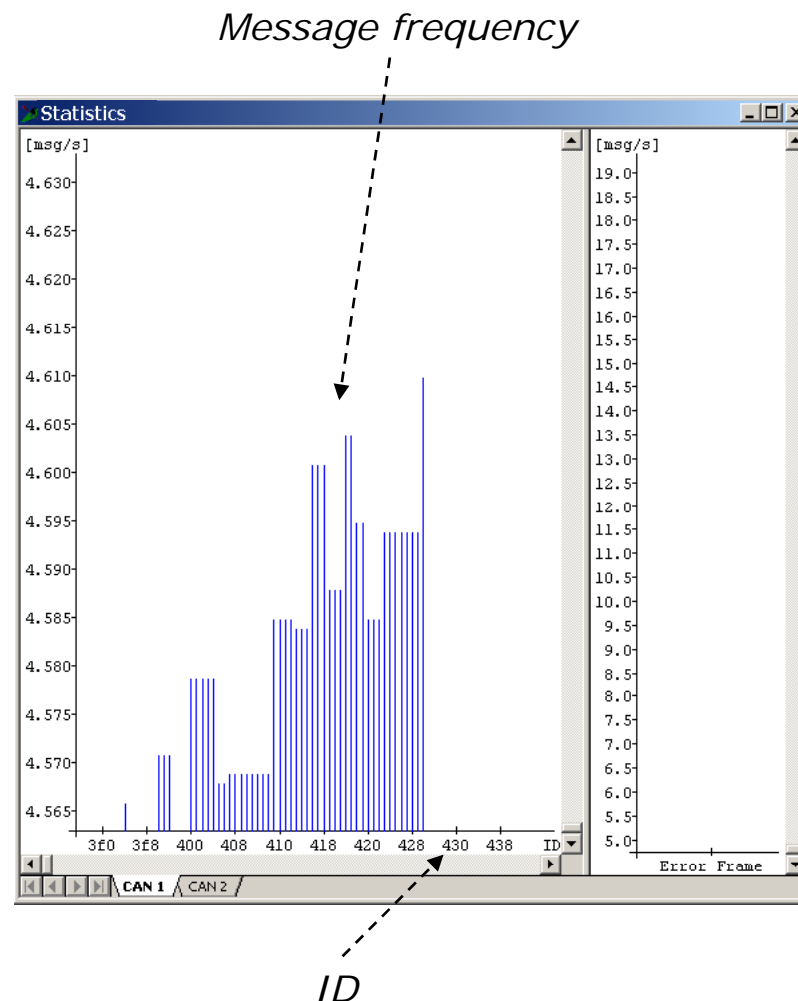
□ 选择需要的信号



CANoe窗口介绍（4）

Statistics Window

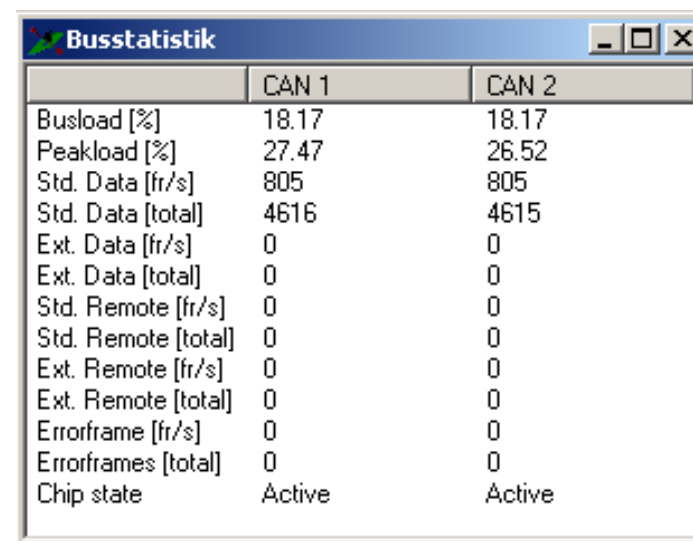
- 显示报文出现频率
- 显示错误帧出现频率
- 统计报告
 - 右键单击空白处
 - Configuration
 - Active
 - 生成统计报告(Write Window)



CANoe窗口介绍（5）

□ Bus Statistics Window

- 总线负载
- 数据帧
- 错误帧
- CAN卡控制器状态



	CAN 1	CAN 2
Busload [%]	18.17	18.17
Peakload [%]	27.47	26.52
Std. Data [fr/s]	805	805
Std. Data [total]	4616	4615
Ext. Data [fr/s]	0	0
Ext. Data [total]	0	0
Std. Remote [fr/s]	0	0
Std. Remote [total]	0	0
Ext. Remote [fr/s]	0	0
Ext. Remote [total]	0	0
Errorframe [fr/s]	0	0
Errorframes [total]	0	0
Chip state	Active	Active

CANoe窗口介绍（6）

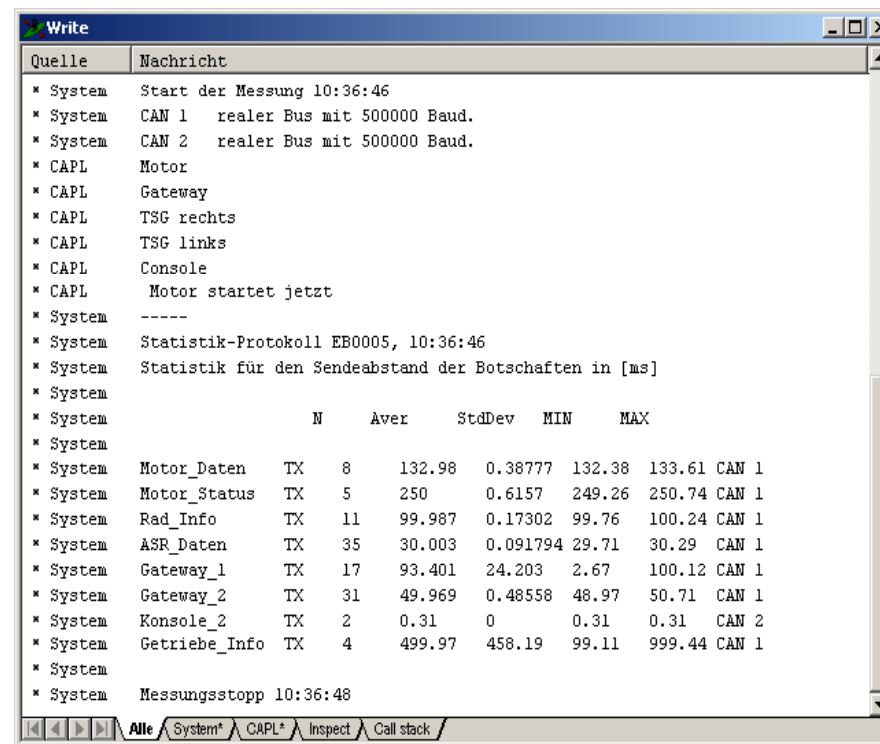
□ Write Window

□ License信息

□ 统计报告

□ CAPL输出窗口

□ Printf = Write

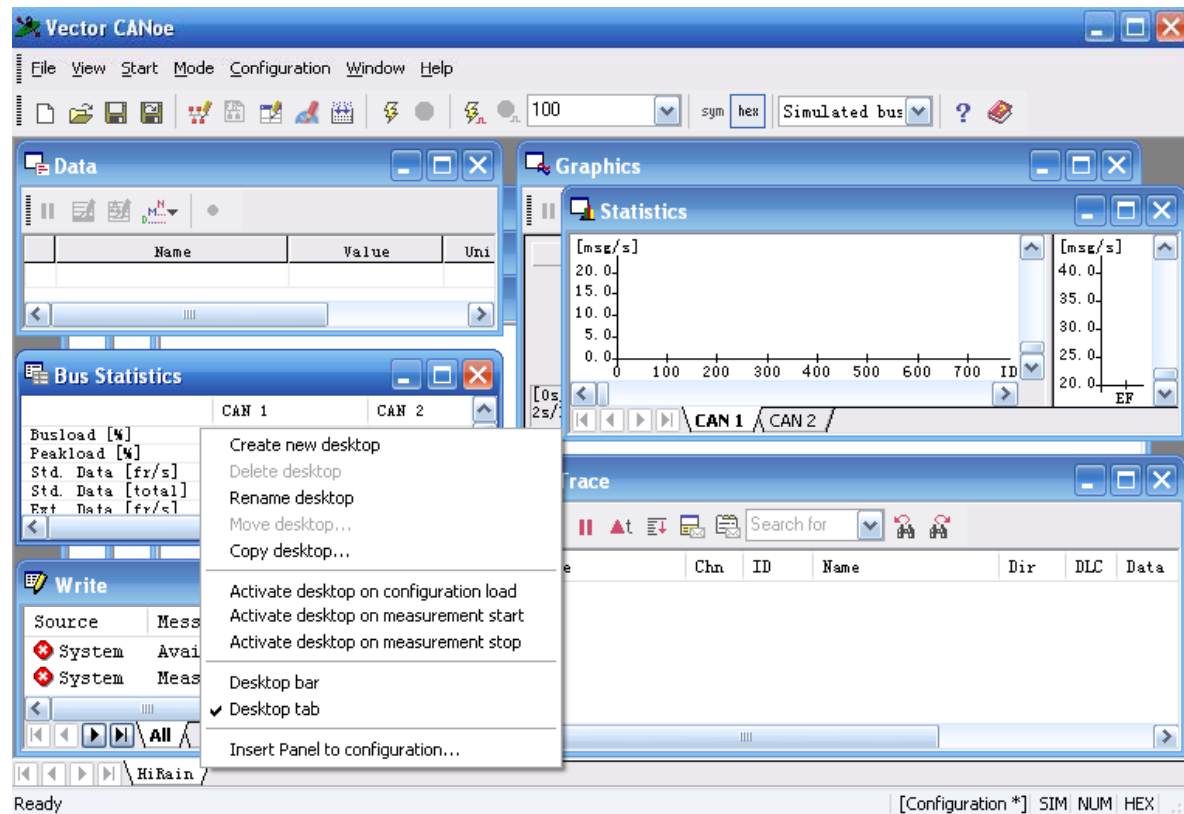


蒙太奇 (3)

□ Desktop

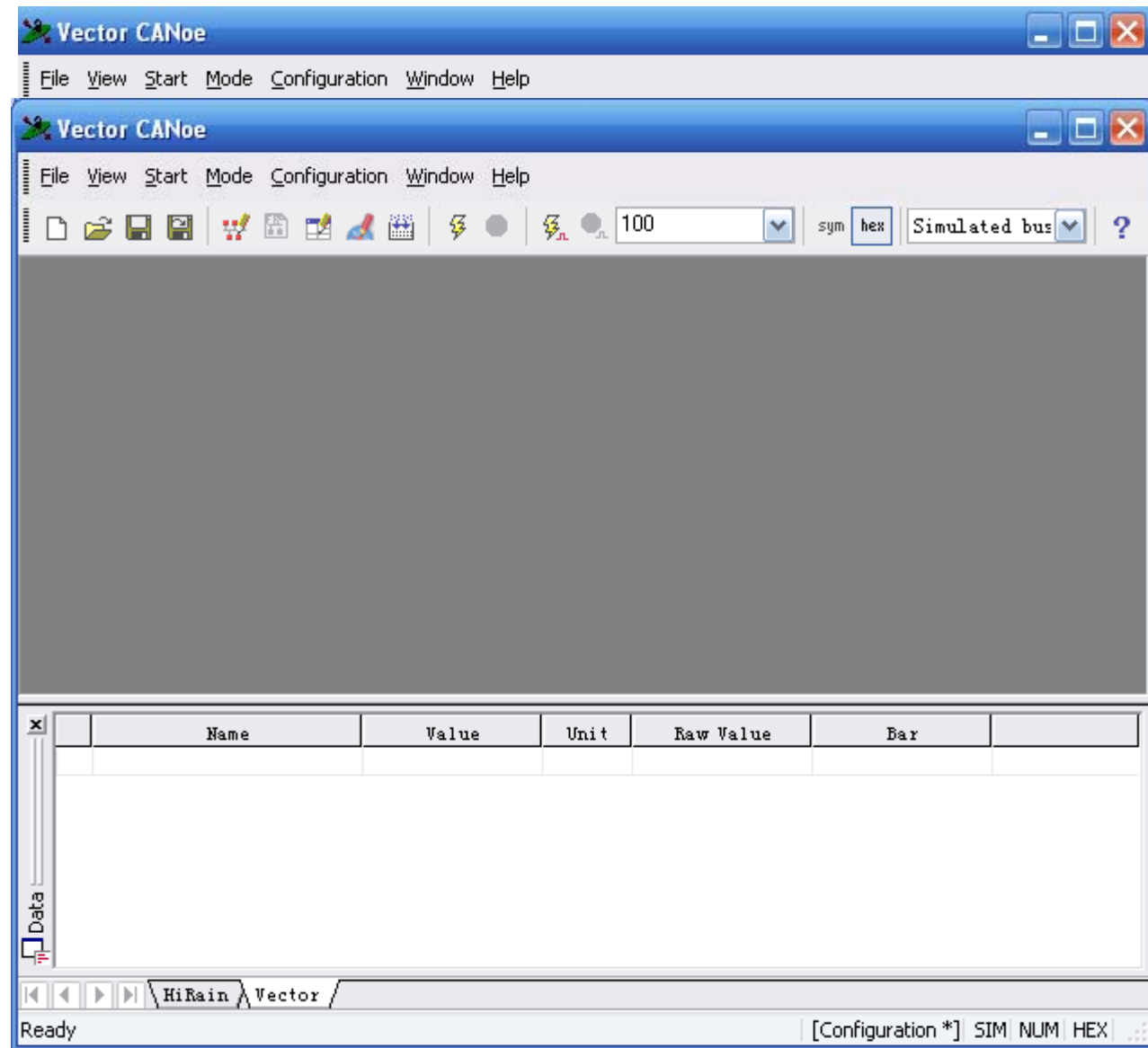
□ Create New Desktop

□ Rename Desktop



蒙太奇（4）

□ 固定窗口

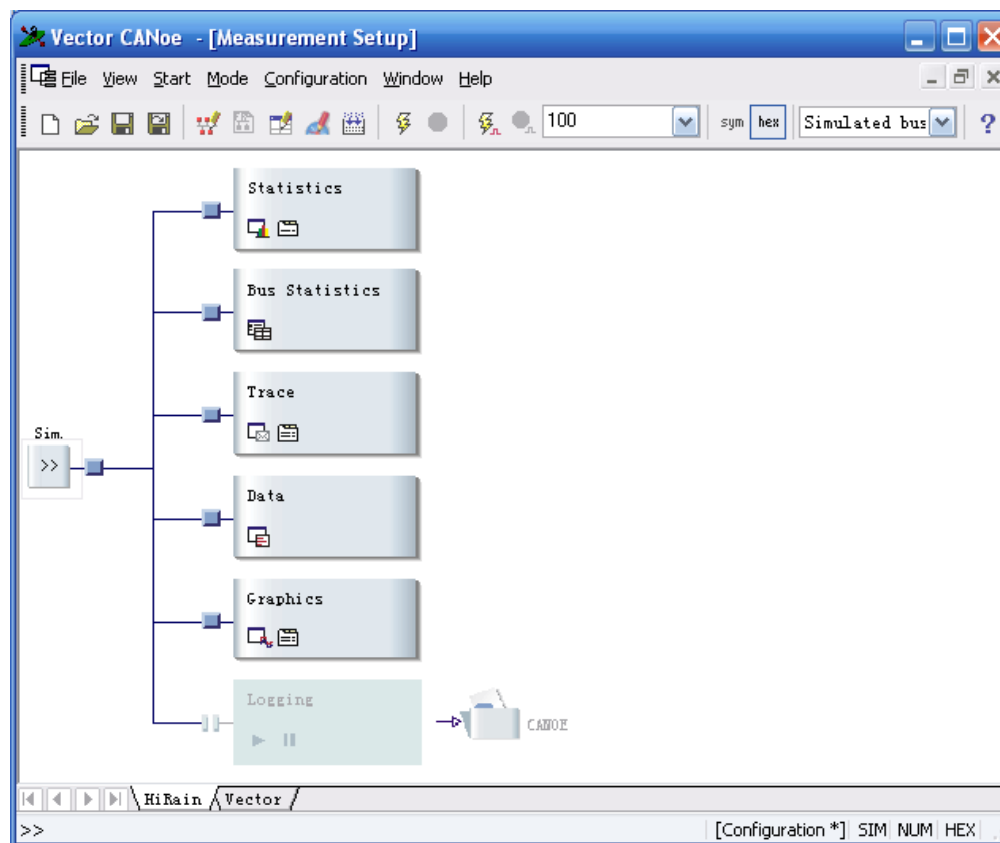


CANoe窗口介绍（7）

□ Measurement Setup

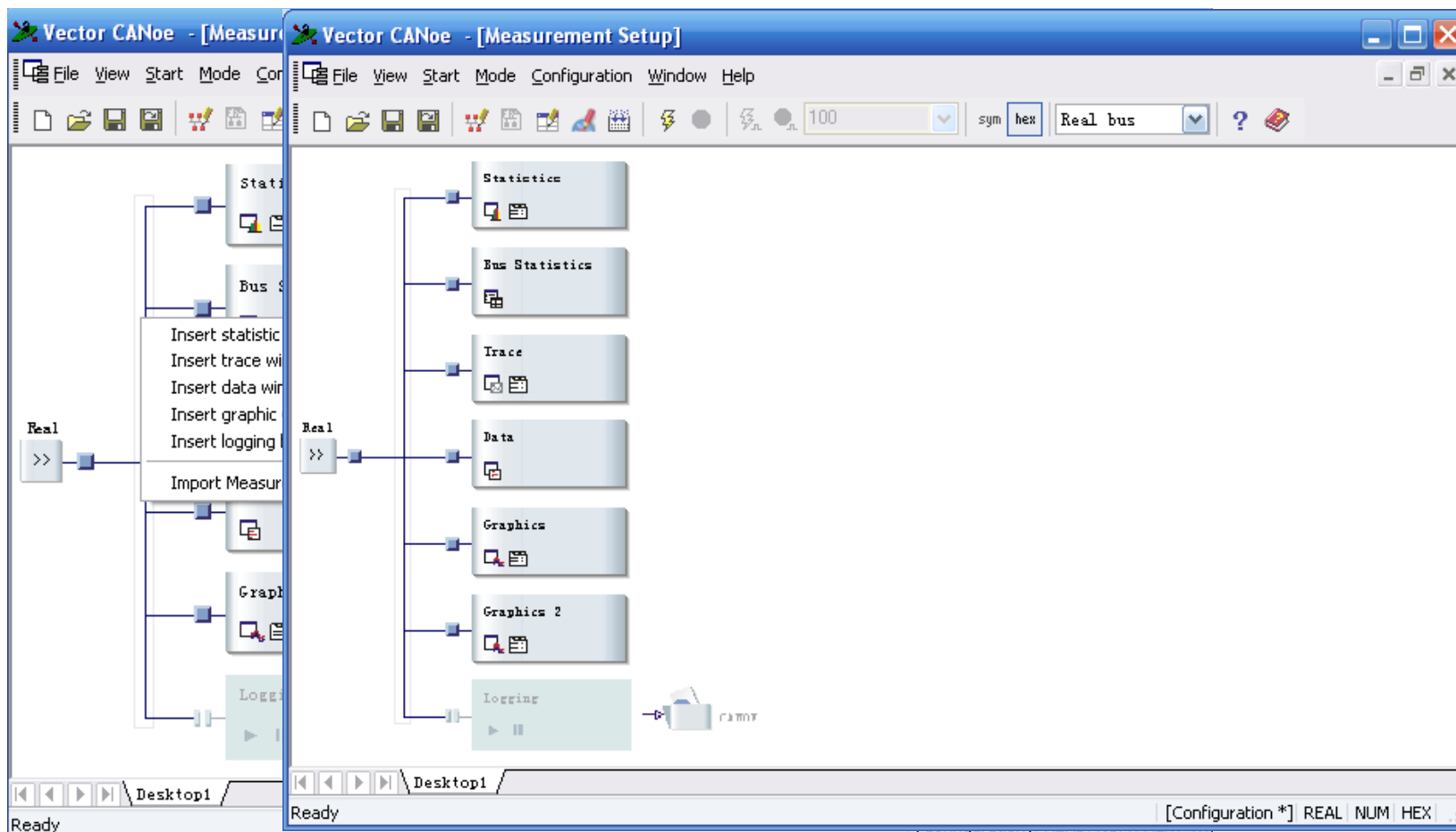
□ View->Measurement Setup

- 每个模块对应一个窗口
- 增加新模块（窗口）
- 插入功能块
- 数据记录



CANoe窗口介绍（7）

▣ 新增模块（窗口）



CANoe窗口介绍（7）

□ 插入功能块

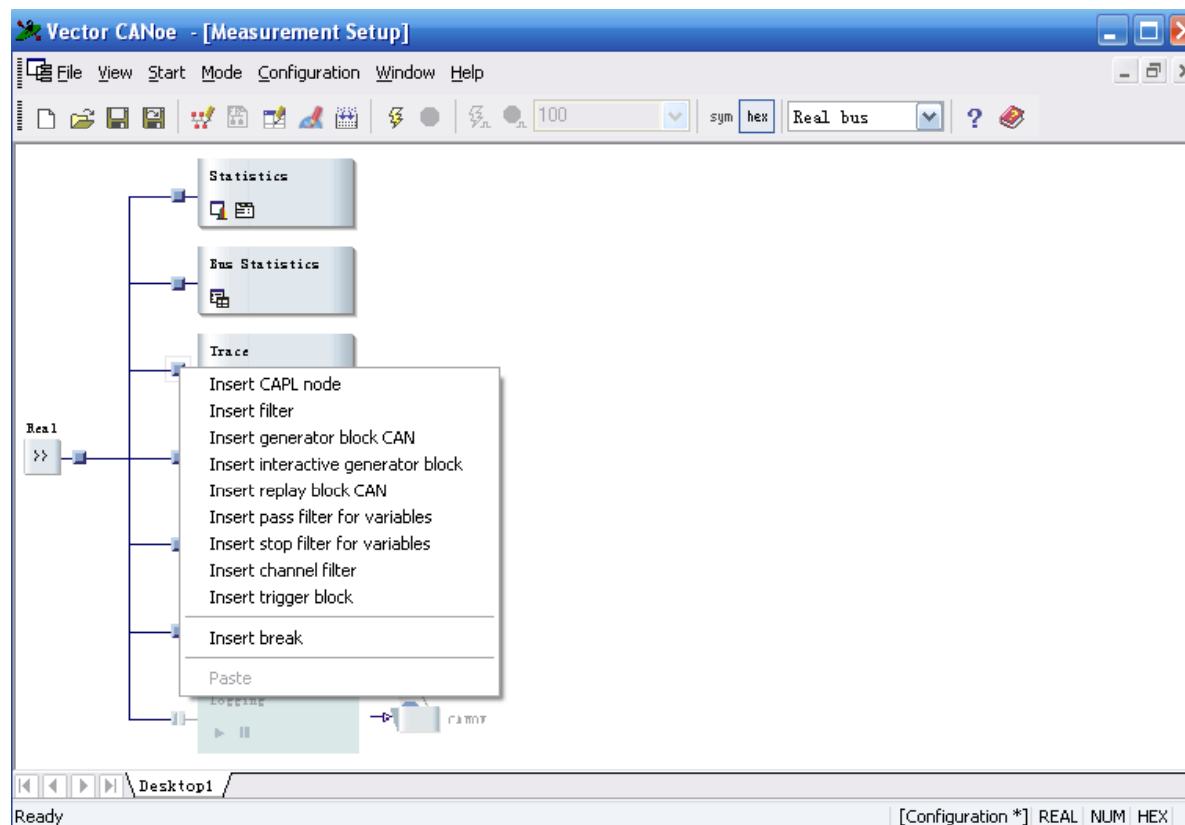
□ CAPL节点

□ 发生器模块

□ 回放模块

□ 触发模块

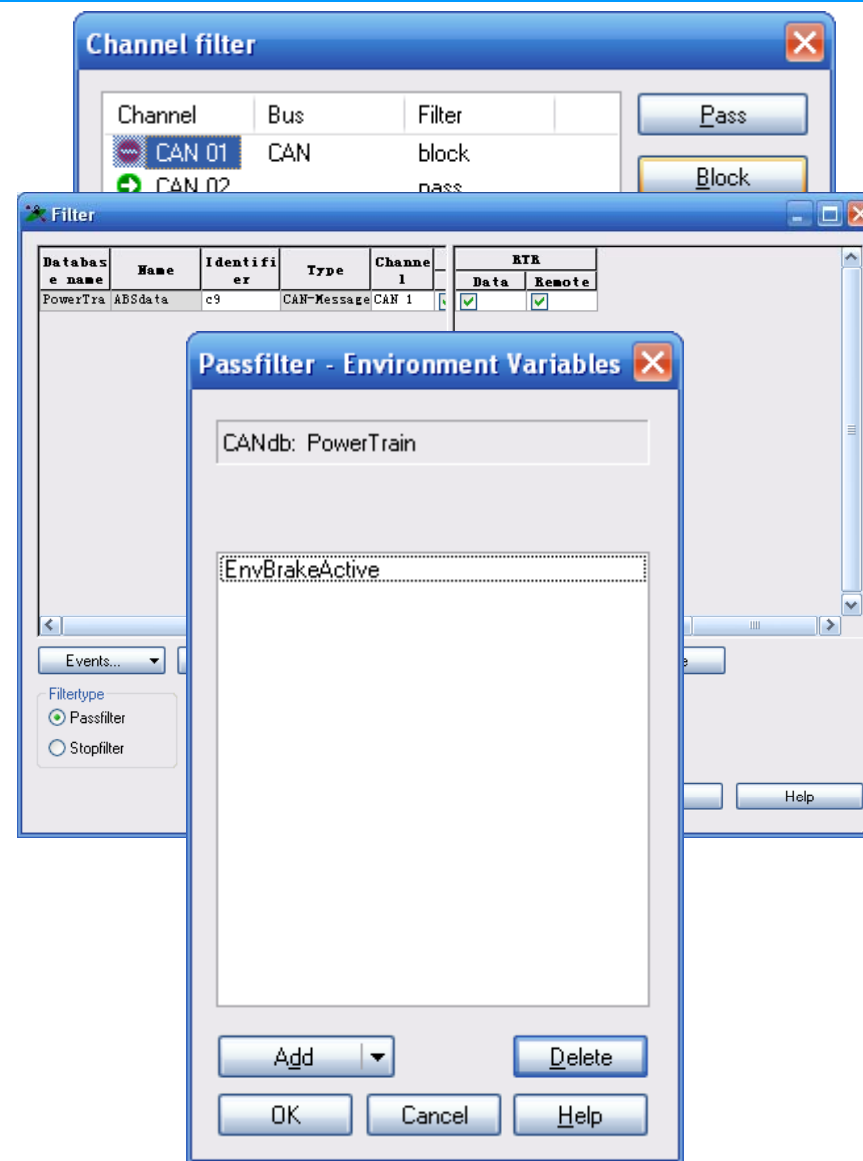
□ 过滤器模块



CANoe窗口介绍（7）

❑ 过滤器模块

- ❑ 通道过滤（Channel Filter）
- ❑ 报文过滤（Filter）
- ❑ 变量过滤（Variables）



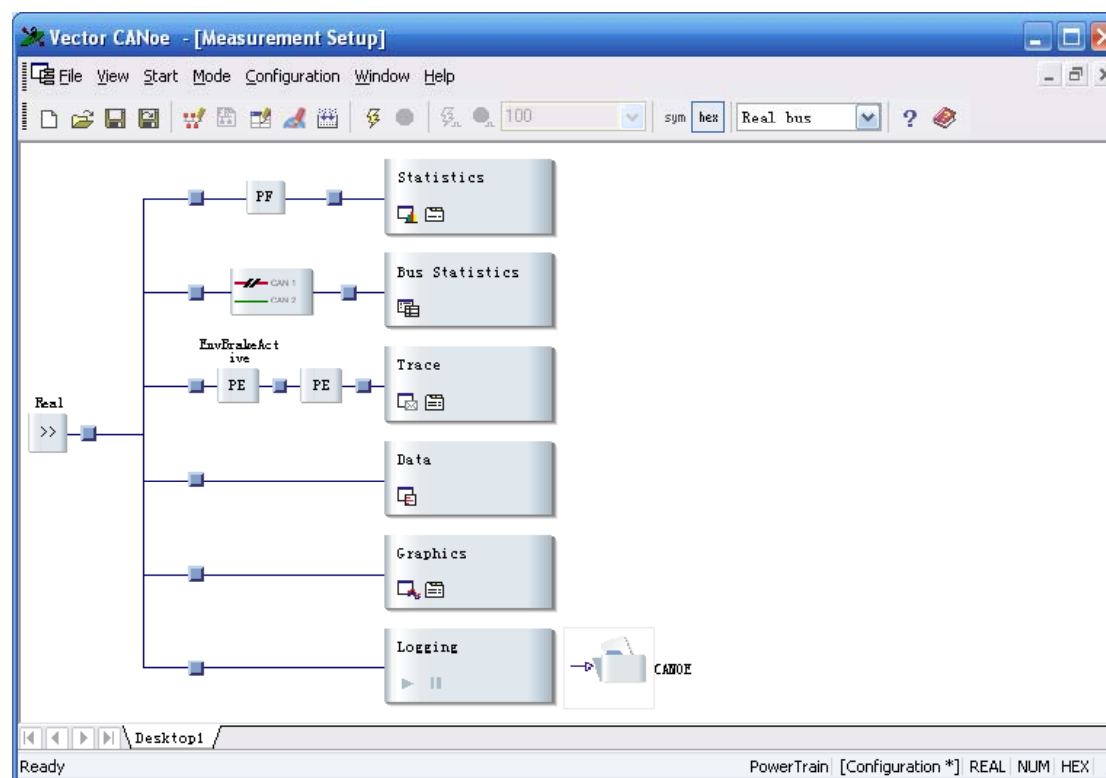
CANoe窗口介绍（7）

□ 数据记录

□ 默认状态关闭

□ 多种记录文件类型

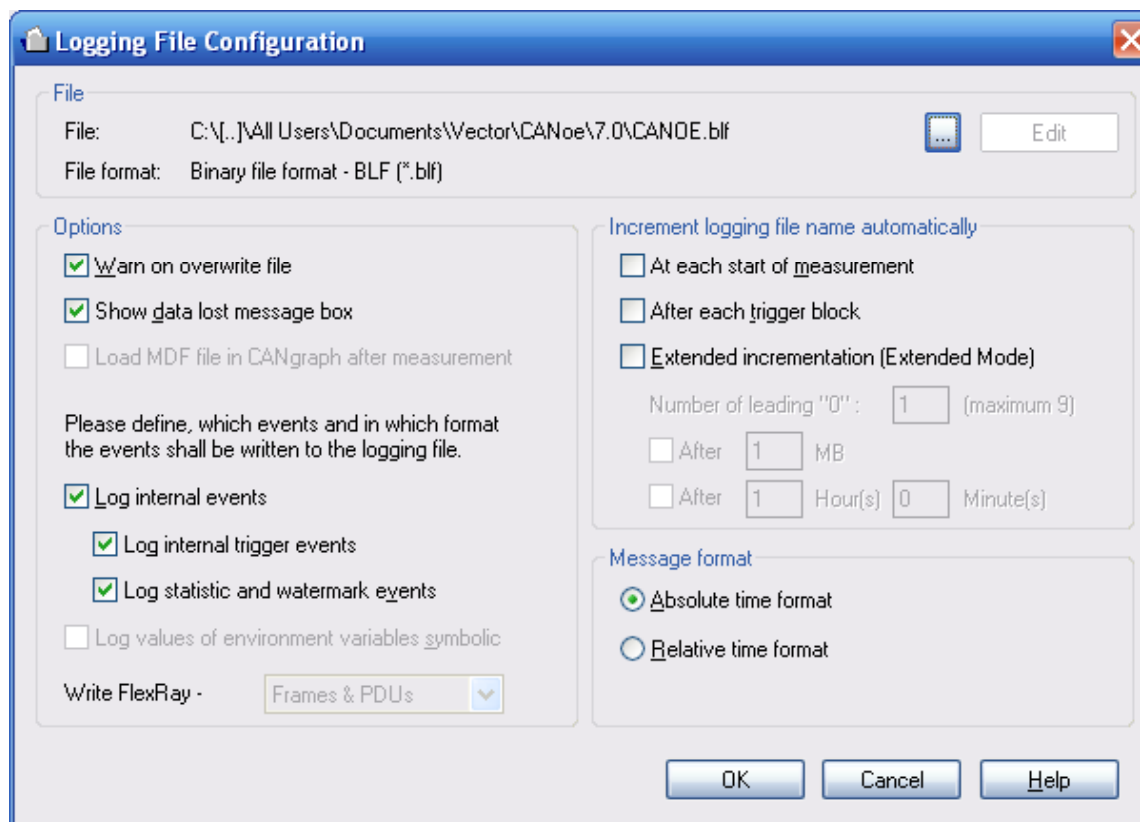
□ 多种记录配置方式



CANoe窗口介绍（7）

□ 记录文件

□ 右键点击文件图标->Logging file configuration



CANoe窗口介绍（7）

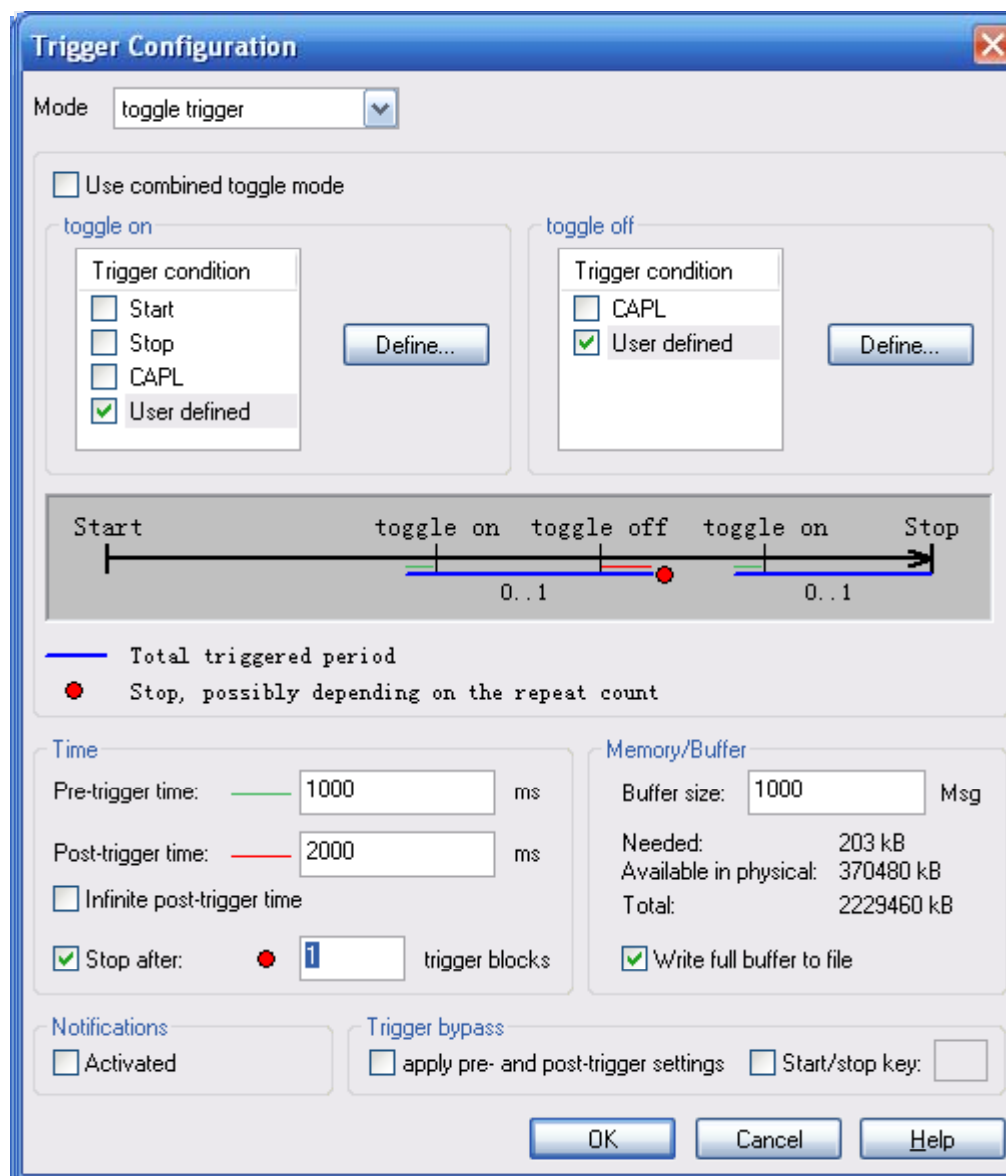
□ 记录配置方式

□ 双击Logging模块

□ 全部记录

□ 单次记录

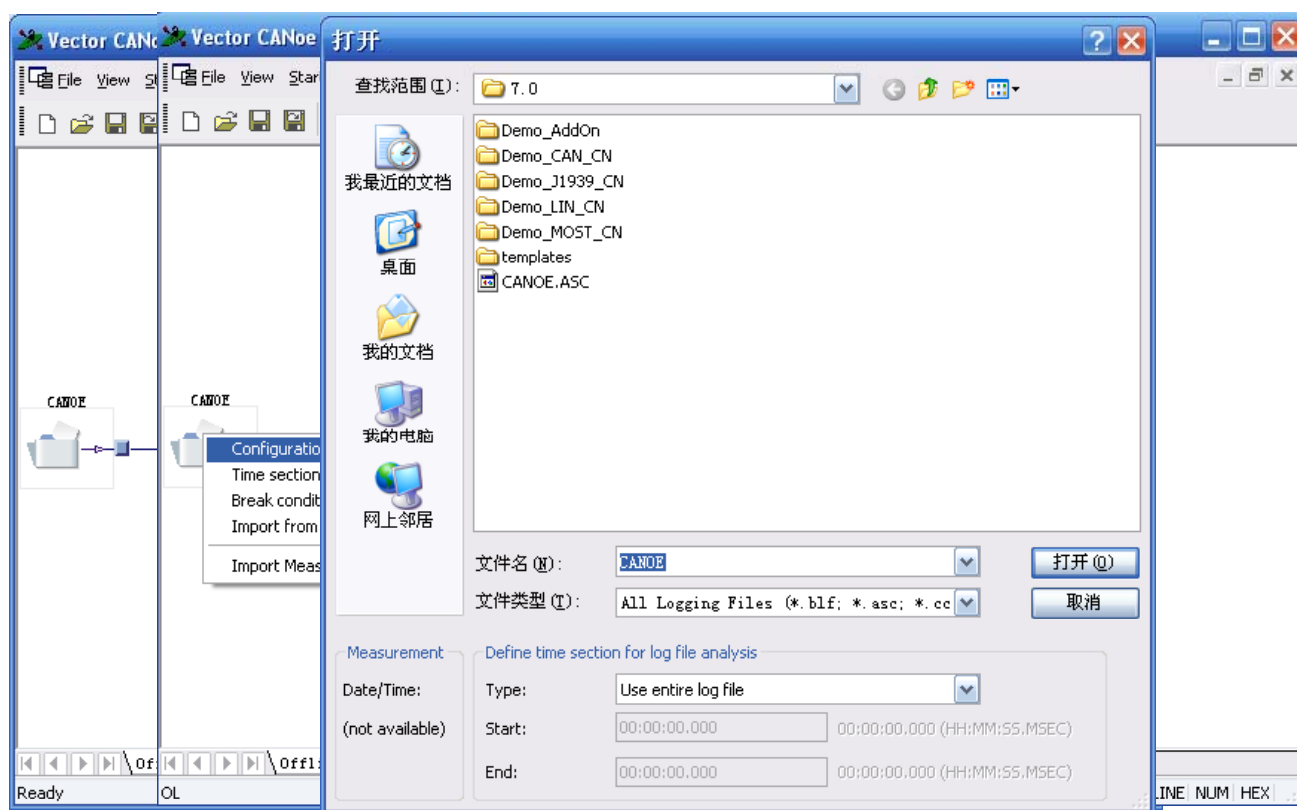
□ 触发记录



蒙太奇 (5)

□ 数据记录的目的是为了离线分析

□ Mode->To Offline

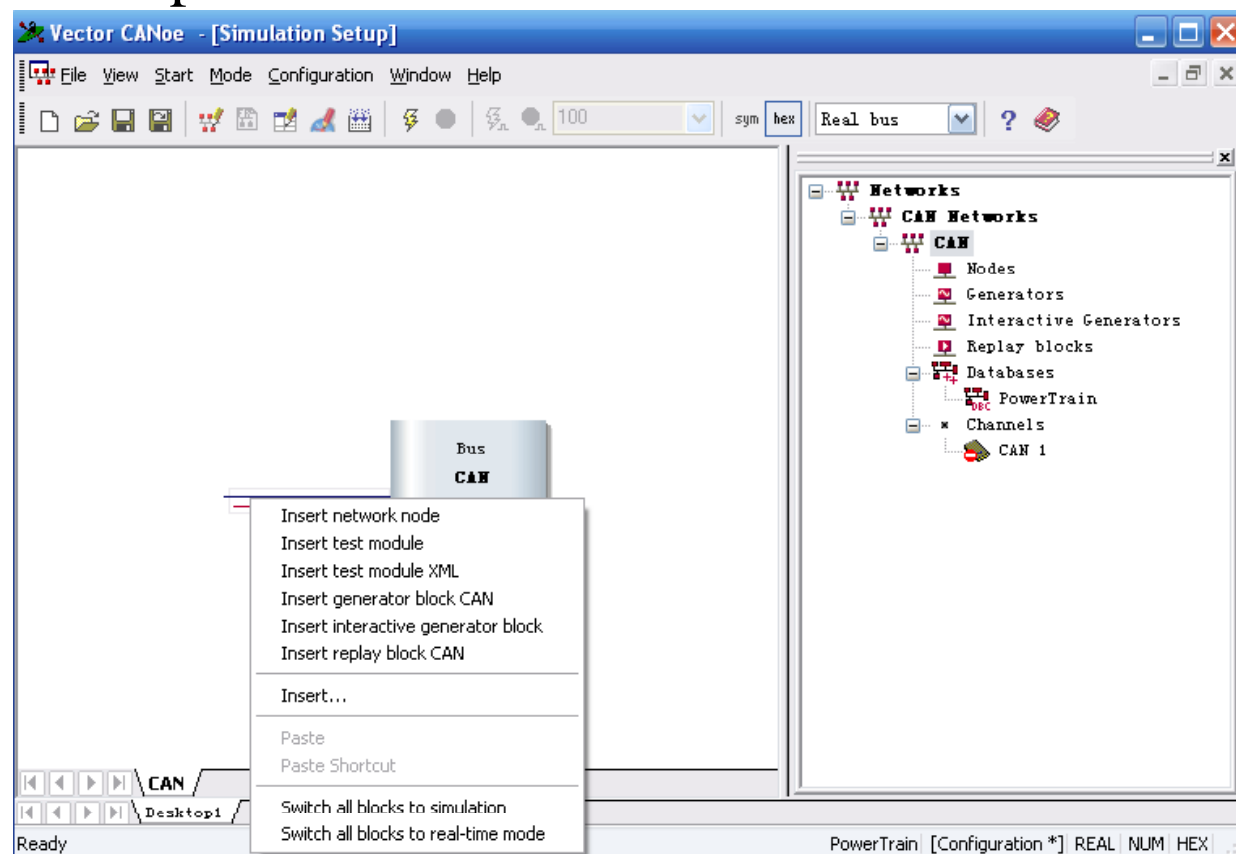


CANoe窗口介绍（8）

□ Simulation Setup

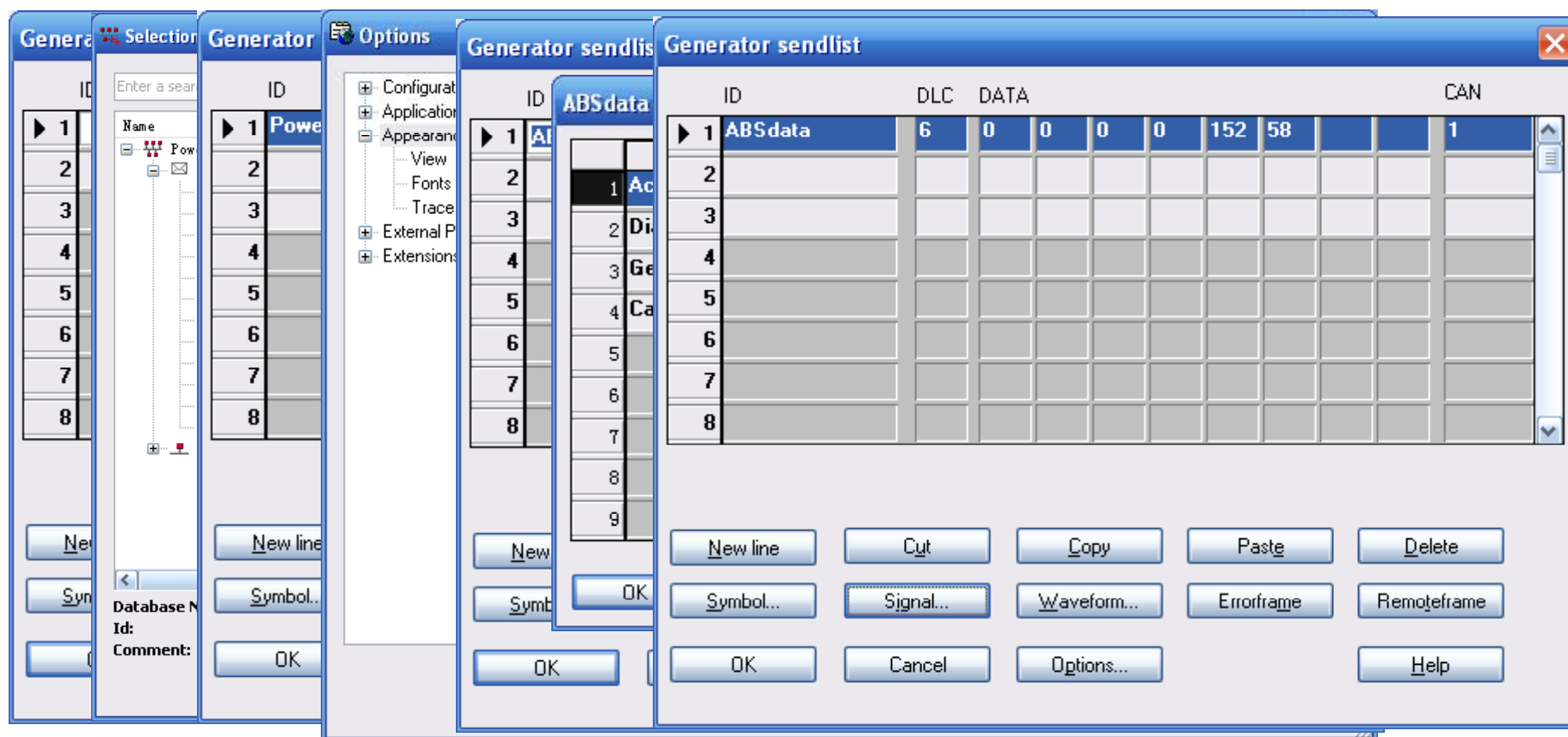
□ View->Simulation Setup

- 发生器
- 交互式发生器
- CAPL节点



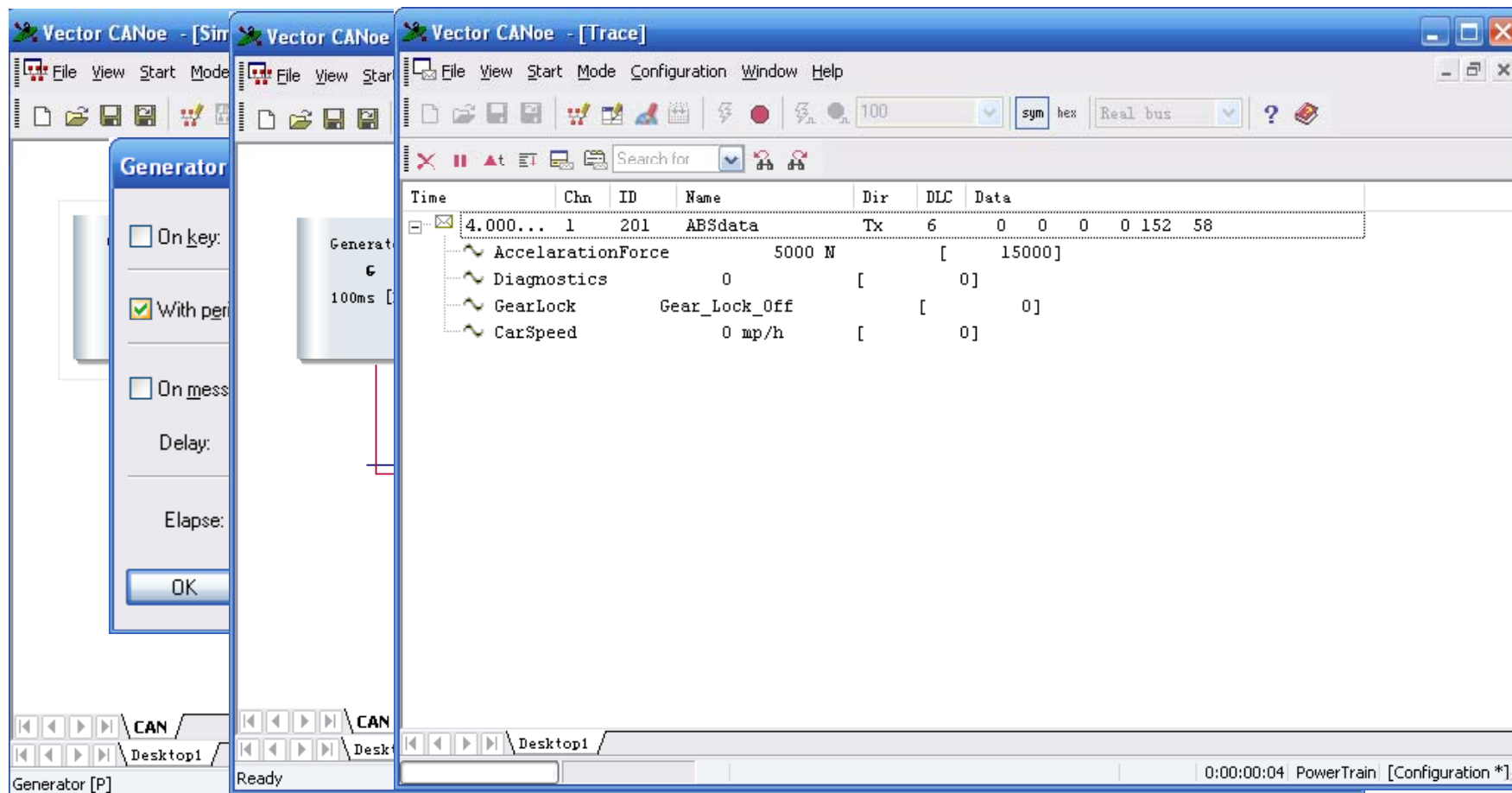
CANoe窗口介绍（8）

□ 发生器模块



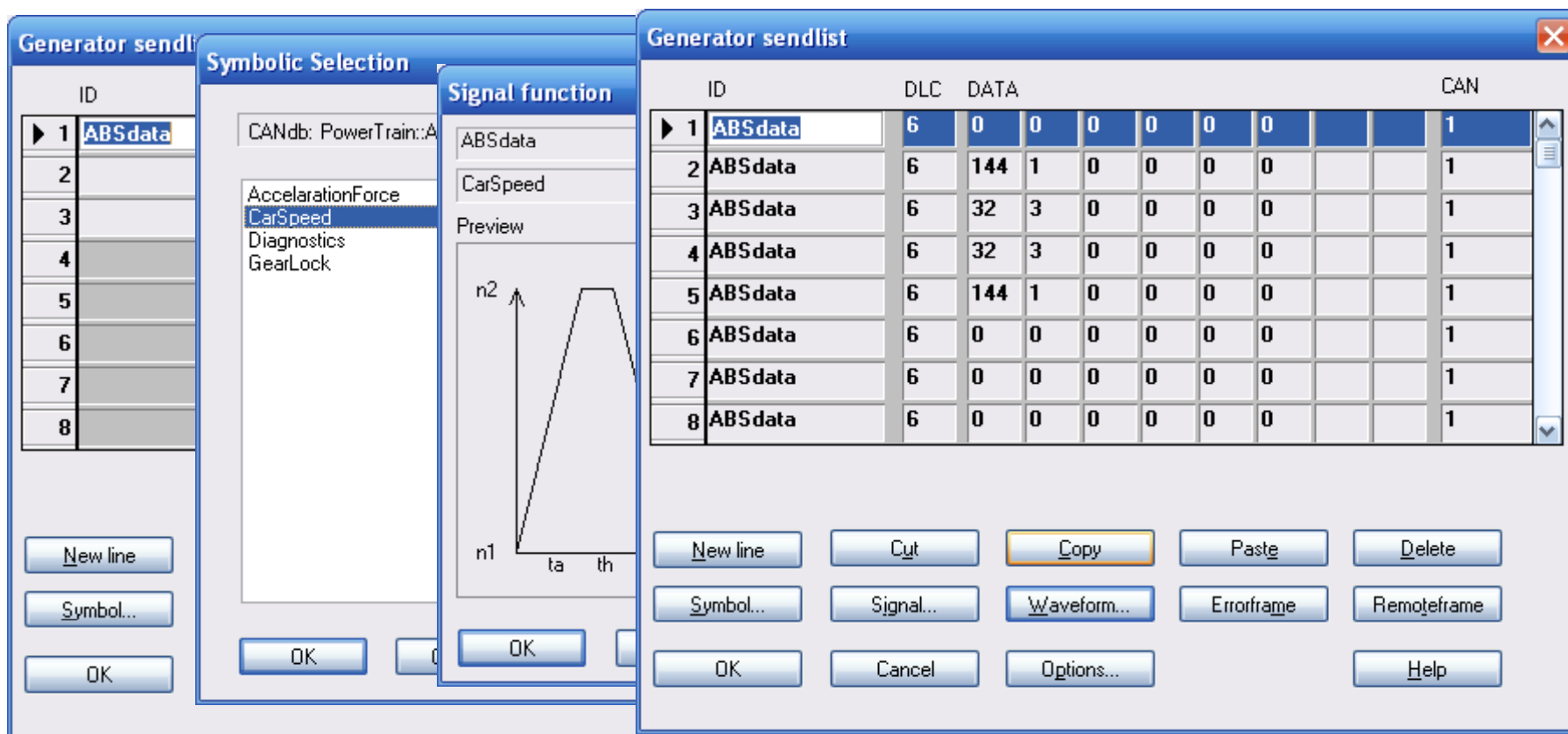
CANoe窗口介绍（8）

□ 发生器模块



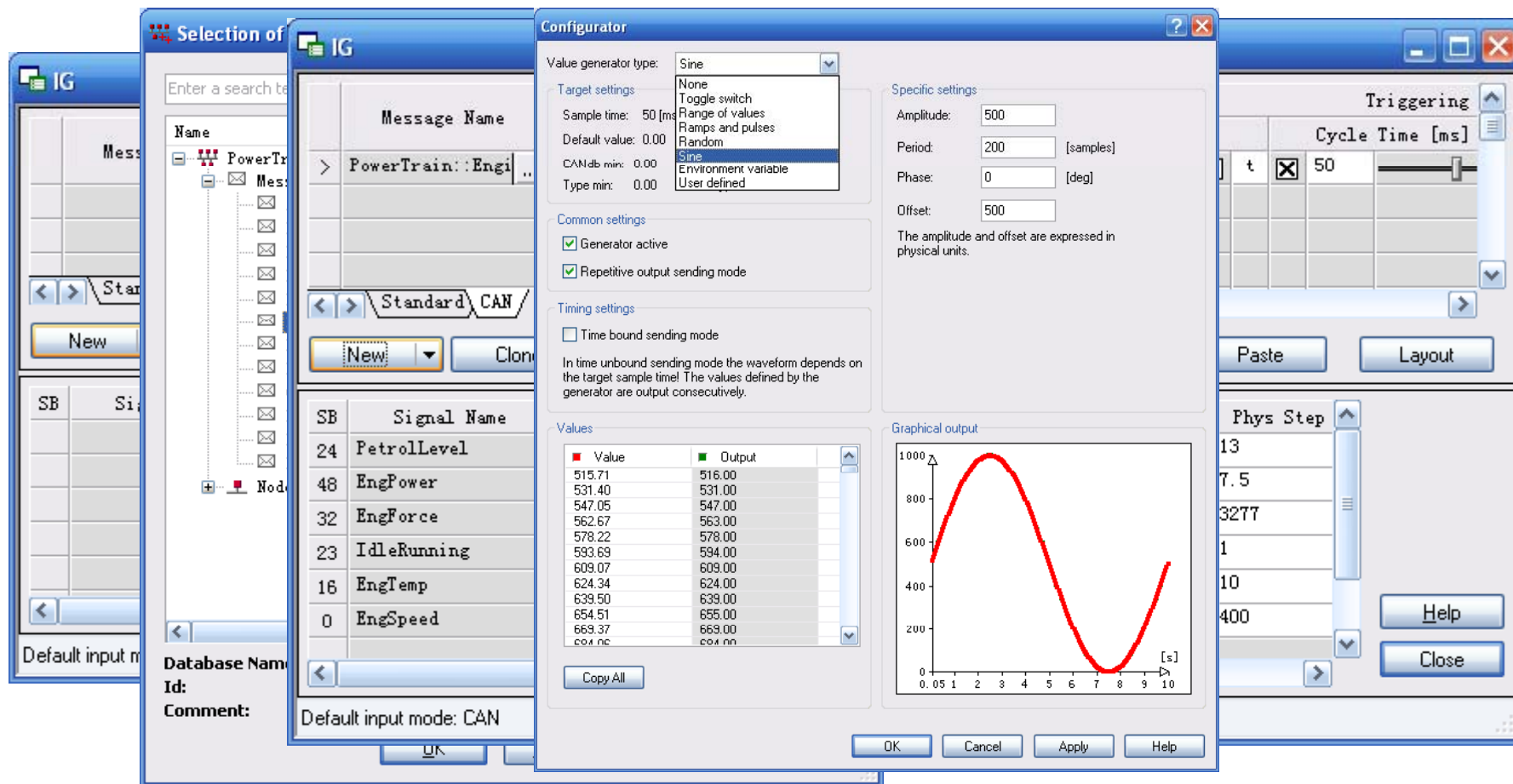
CANoe窗口介绍（8）

□ 发生器模块



CANoe窗口介绍（8）

交互式发生器模块



欢迎进入CAPL的世界

□ CAPL (CAN Access Programming Language)

□ 类C语言

□ 仿真

□ 单个节点和整个网络

□ 外部环境

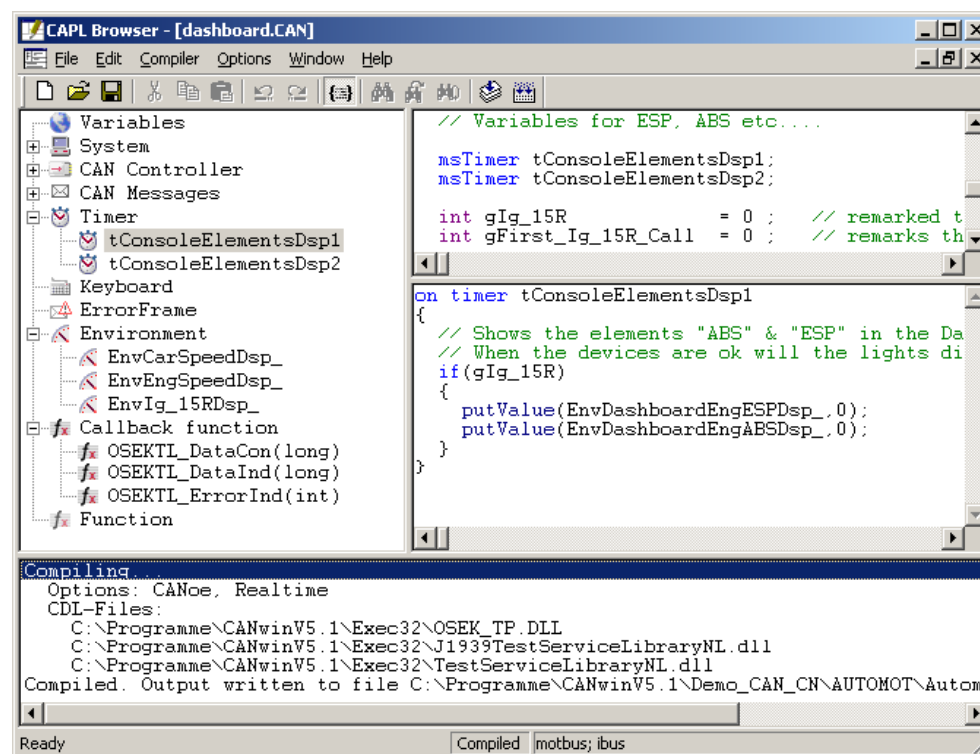
□ 测试

□ 面向事件的编程语言

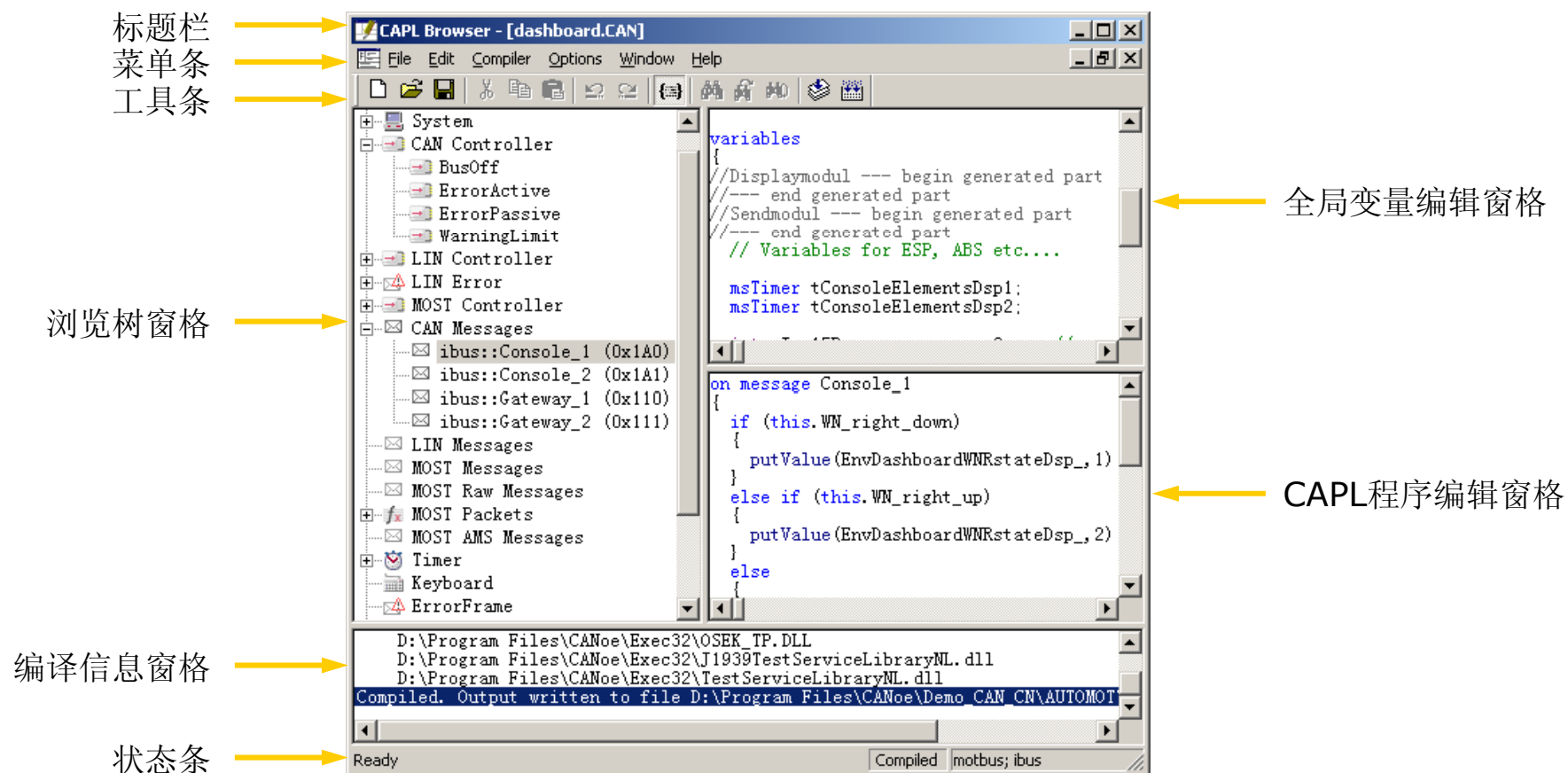
□ 总线事件

□ 键盘事件

□ 时间事件



CAPL Browser



CAPL事件

事件类型	事件名	程序执行条件	事件过程语法结构 *
系统事件	<i>PreStart</i>	CANoe初始化时执行	<i>on preStart { ... }</i>
	<i>Start</i>	测量开始时执行	<i>on start { ... }</i>
	<i>StopMeasurement</i>	测量结束时执行	<i>on stopMeasurement { ... }</i>
CAN控制器事件	<i>BusOff</i>	硬件检测到BusOff时执行	<i>on busOff { ... }</i>
	<i>ErrorActive</i>	硬件检测到ErrorActive时执行	<i>on errorActive { ... }</i>
	<i>ErrorPassive</i>	硬件检测到ErrorPassive时执行	<i>on errorPassive { ... }</i>
	<i>WarningLimit</i>	硬件检测到WarningLimit时执行	<i>on warningLimit { ... }</i>
CAN消息事件	自定义	接收到指定的消息时执行	<i>on message Message { ... }</i>
时间事件	自定义	定时时间朝过时执行	<i>on timer Timer { ... }</i>
键盘事件	自定义键值	指定的键被下时执行	<i>on key Key { ... }</i>
错误帧事件	<i>ErrorFrame</i>	硬件每次检测到错误帧时执行	<i>on errorFrame { ... }</i>
环境变量事件	自定义	指定的环境变量值改变时执行	<i>on envVar EnvVar { ... }</i>

CAPL基本语法

□ 类C语言，语法与C语言基本相同

□ 注释

□ // 放置在需要注释的语句之前，注释单行

□ /* 注释起始符，其后的内容被注释

□ */ 注释结束符，结束由 ‘/*’开始的注释

□ 分号 程序结束标识

□ 大括号 函数体

```
counter = counter+1;
if (counter==256)
{
    counter=0;
    stop();
}
```

消息事件

- on message 123 //对消息123(dec)反应
- on message 0x123 //对消息123(hex)反应
- on message MotorData //对消息MotorData(符号名字)反应
- on message CAN1.123 //对CAN 通道1收到消息123反应
- on message * //对所有消息反应
- on message 100-200 //对100-200间消息反应

键盘事件

- on key 'a' //按 'a'键反应
- on key ' ' //按空格键反应
- on key 0x20 //按空格键反应
- on key F1 //按F1键反应
- on key Ctrl-F12 //按Ctrl + F12键反应
- on key PageUP //按PageUp键反应
- on key Home //按Home键反应
- on key * //按所有键反应

时间事件

□ 定时器声明

□ `msTimer myTimer;` //将myTimer 申明ms为单位的变量

□ `timer myTimer;` //将myTimer 申明s为单位的变量

□ 定时器函数

□ `setTimer(myTimer,20);` //将定时值设定为20ms，并启动

□ `cancelTimer(myTimer);` //停止定时器myTimer

□ 定时器事件

□ `on timer myTimer` //对myTimer 设定的时间到反应

环境变量事件

□ 环境变量函数

□ `getValue()` //获取环境变量的值

□ `putValue()` //设置环境变量的值

□ 环境变量事件

□ `on envVar XXX`

数据类型

数据类型	名称	注释
无符号整型	byte	1个字节
	word	2个字节
	dword	4个字节
有符号整型	int	2个字节
	long	4个字节
浮点型	float	8个字节
	double	8个字节
CAN报文	message	
定时器	timer	秒
	msTimer	毫秒
单个字符	char	1个字节

数据定义

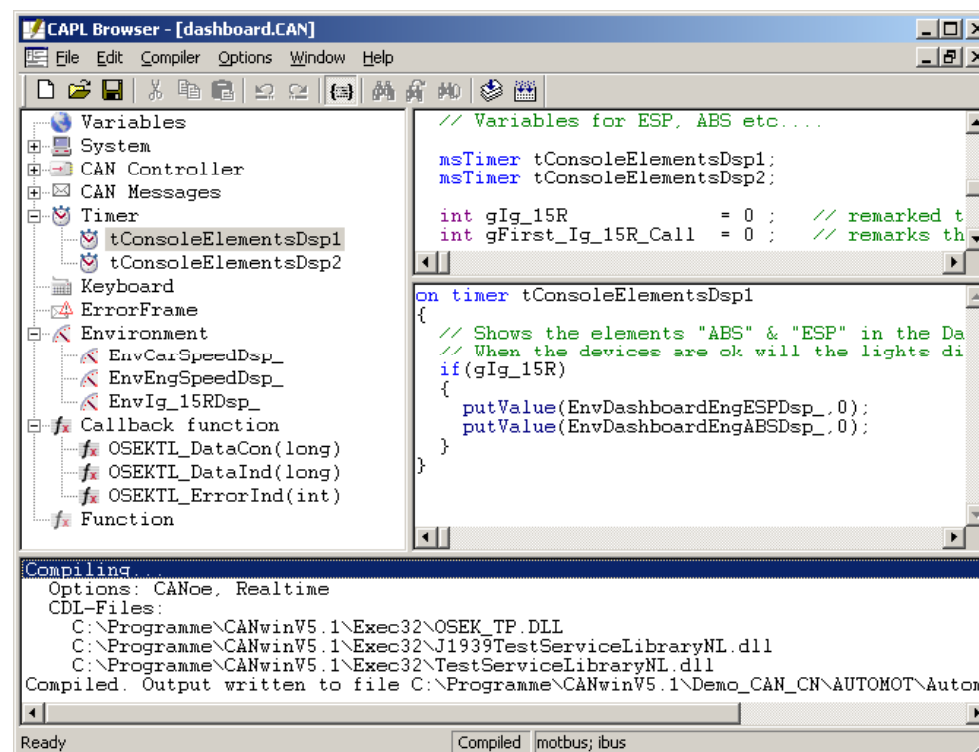
▣ 全局变量和局部变量

▣ 变量定义

`int i;`

`message 0x123 HiRain;`

`message MotorData Vector;`



完整的CAPL程序

□ 三个部分

□ 变量

□ 各种事件

□ 自定义函数

```
variables
{
    ...           //申明全局变量
}
```

```
on start
{
    ...           //过程指令块
}

on message xxx
{
    ...           //过程指令块
}

on key '1'
{
    ...           //过程指令块
}
```

```
My_function_1(Para_1, Para_2, ...)
{
    ...           //函数体
}

...

My_function_n(Para_1, Para_2, ...)
{
    ...           //函数体
}
```

CAPL输出文本

□ Write Window

□ write函数

```
int h=100;  
char ch='a';  
char s100[8]="hundred";  
write("Hundred as a number:%d,%x",h,h);  
write("Hundred as a string:%s",s100);  
write("The square root of two is %6.4g",sqrt(2.0));
```

消息处理常用语句

- `if (this.id==100) { ... }`
- `msg.can=2;`
- `msg.dlc=8;`
- `dword t ; t=this.time;`
- `if(this.dir!=RX) { return; }`
- `this.CarSpeed = 200;`

关键字this

□ this代表触发事件的对象

```
on message 100 {  
    byte byte_0;  
    byte_0 = this.byte(0);  
    ...  
}
```

```
on envVar Switch {  
    int val;  
    val = getvalue(this);  
    ...  
}
```

报文处理

on message 0x64

```
{  
    if(this.byte(2)==0xFF)  
        write("Third byte of the message is invalid");  
}
```

on message MotorData

```
{  
    if(this.temperature.phys>=150)  
        write("Warning: critical temperature");  
}
```

键盘处理

```
on key 'a' {  
    message MotorData mMoDa;  
    mMoDa.temperature.phys=60;  
    mMoDa.speed.phys=4300;  
    output(mMoDa);  
}  
on key 'b' {  
    message 100 m100= { dlc=1 };  
    m100.byte(0)=0x0B;  
    output(m100);  
}
```

定时器处理

Variables

```
{  
    message 0x555 msg1 = { dlc=1 };  
    msTimer timer1;  
}
```

on start

```
{  
    setTimer(timer1,100);  
}
```

on timer timer1

```
{  
    setTimer(timer1,100);  
    msg1.byte(0)=msg1.byte(0)+1;  
    output(msg1);  
}
```

环境变量处理

```
on envVar evSwitch
{
    message MotorData msg;
    msg.bsSwitch = getValue(this);
    output(msg);
}
```

练习1

- 当CANoe启动时，向Write Window输出一句话，例如“Hello the world!”

练习2

- 利用发生器模块周期性发送某一报文，例如每隔200ms发送一条EngineData报文。每当按下a键，在Write Window窗口输出一句话，例如“XXX EngineData messages have sent.”
- 注：XXX为已经发送的EngineData报文数量。

练习3

- 不用发生器模块实现Enginedata报文的周期性发送。
- 每当按下a键时， EngineData里面EngSpeed信号值为2000； 当按下b键时， EngineData里面EngSpeed信号值为4000；
- 如果EngineData里面EngSpeed信号为4000， 则发送ABSDData报文， 同时在Write Window输出“Warning!”
- 当按下c键时， 停止EngineData报文发送。

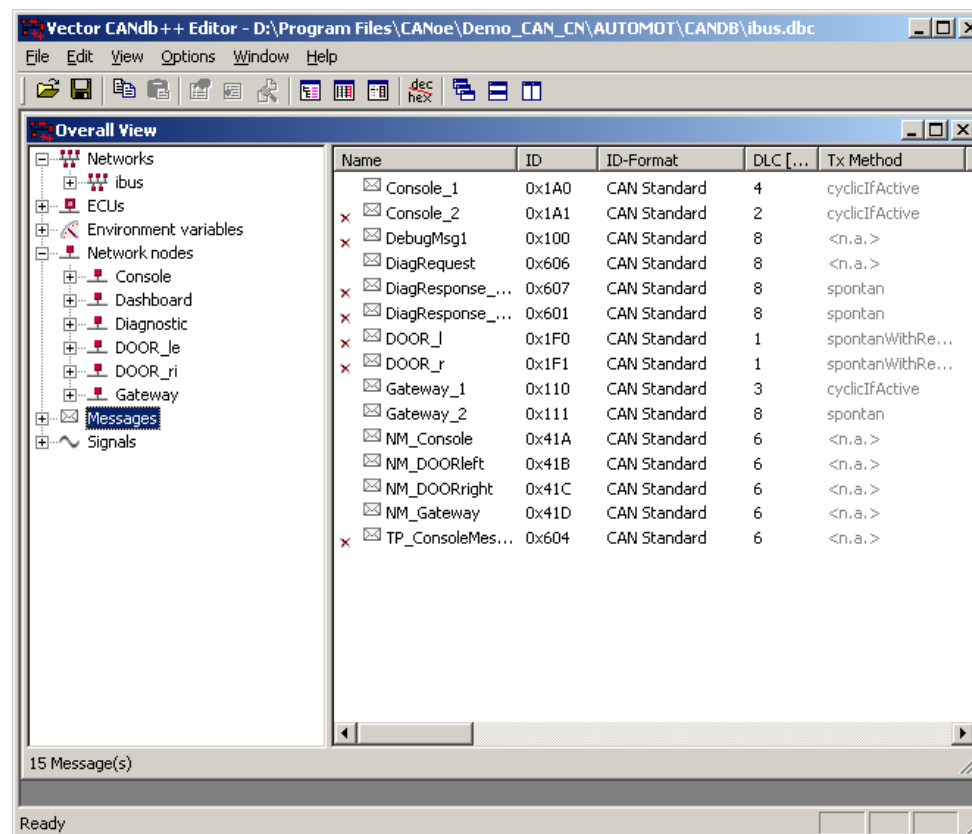
欢迎进入CANdb++ Editor的世界

▣ DBC文件编辑工具

▣ 启动CANoe

▣ File->Open CANdb Editor

▣ 点击



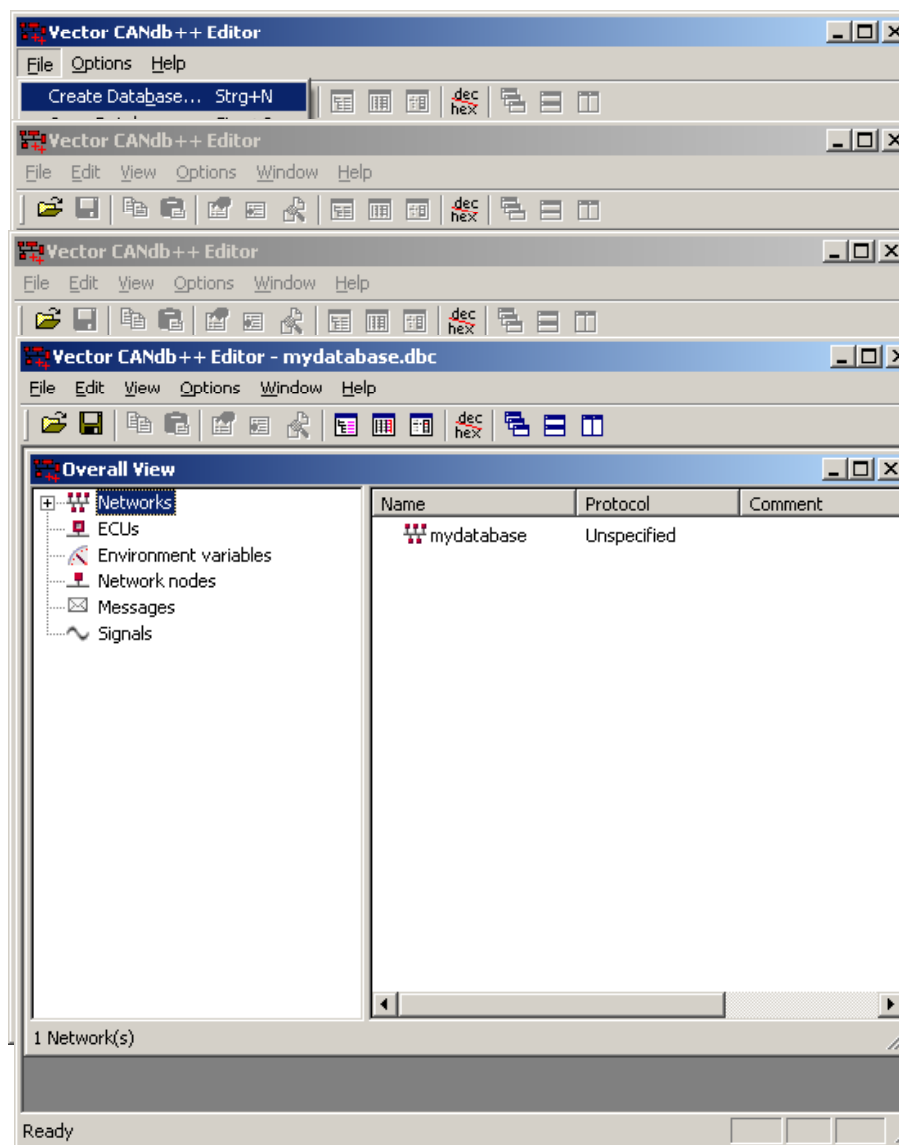
创建一个新的CAN数据库

□ File->Create Database...

□ 选择模板，鼠标双击或按 **[OK]**按钮

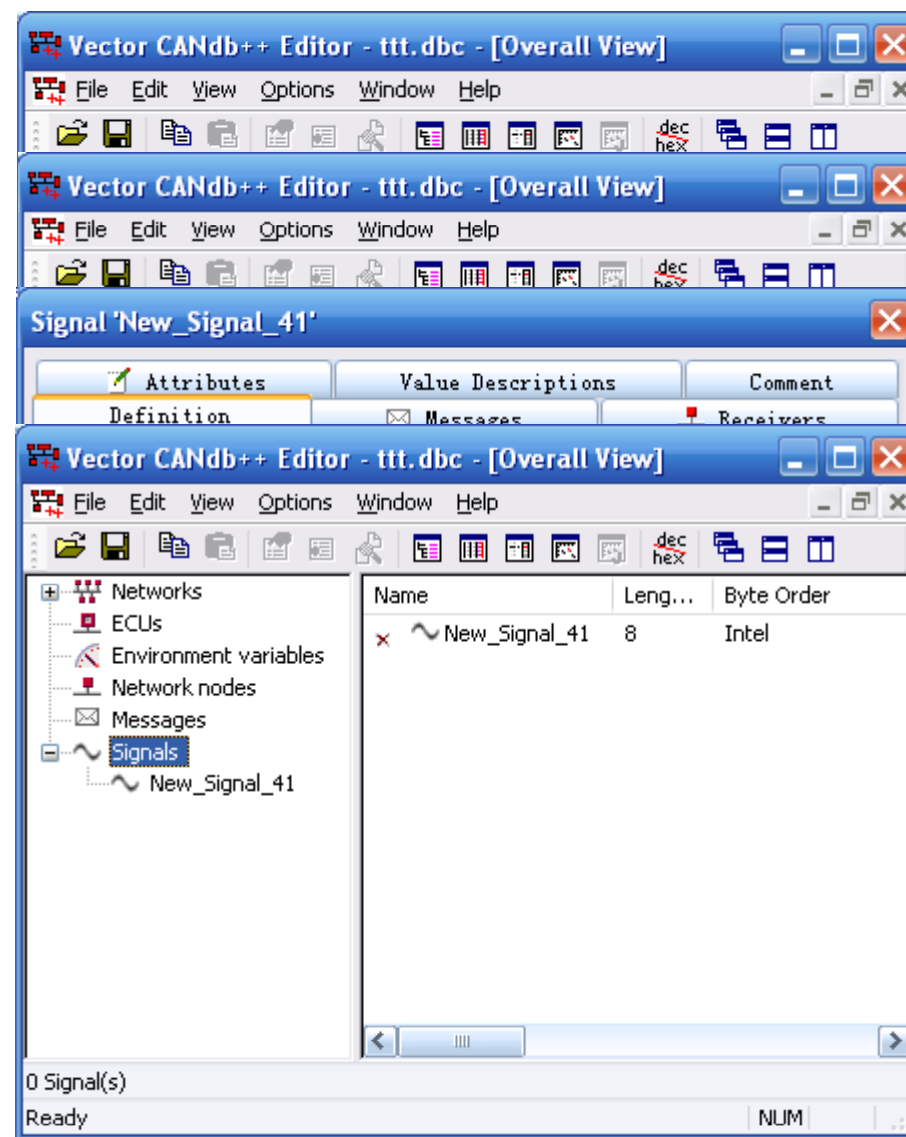
□ 指定数据库文件类型、文件名及保存目录

□ 按**[Save]**按钮。
一个新数据库创建完成



创建对象（信号、报文、节点、环境变量和ECU）

- ❑ 在Overview窗口左边
选择所需创建对象的类型
- ❑ 右键点击对象类型，
在快捷菜单中选择New...
- ❑ 使用配置对话框设置
所创建对象的系统参数值
- ❑ 点击[确定]按钮，
一个新对象便创建完毕

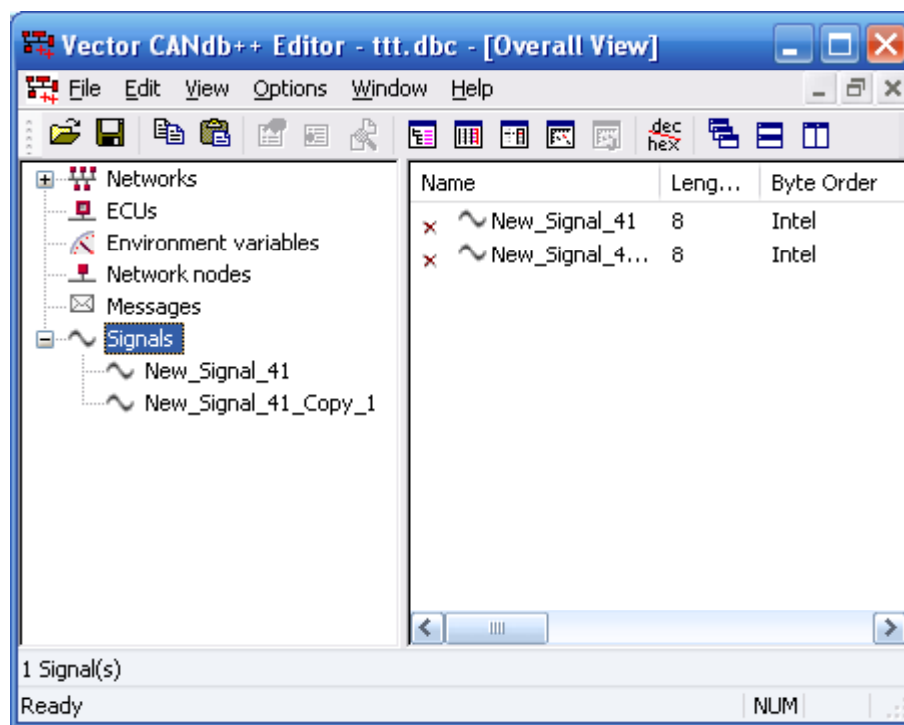


复制已有对象

□ Copy-Paste

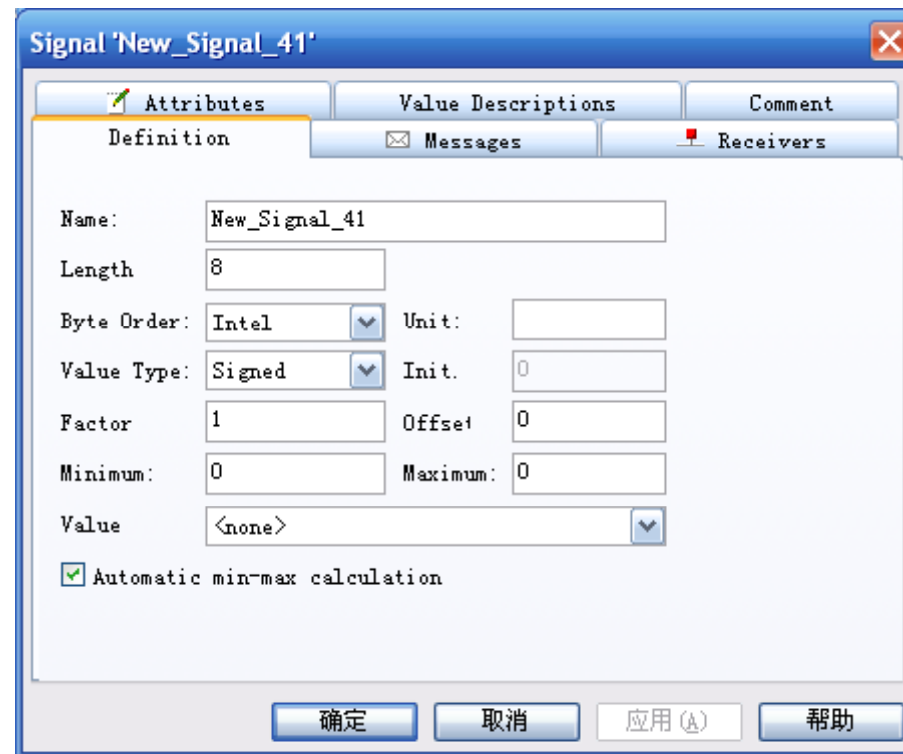
□ 选择已有对象Ctrl+c

□ 选择对象类型Ctrl+v



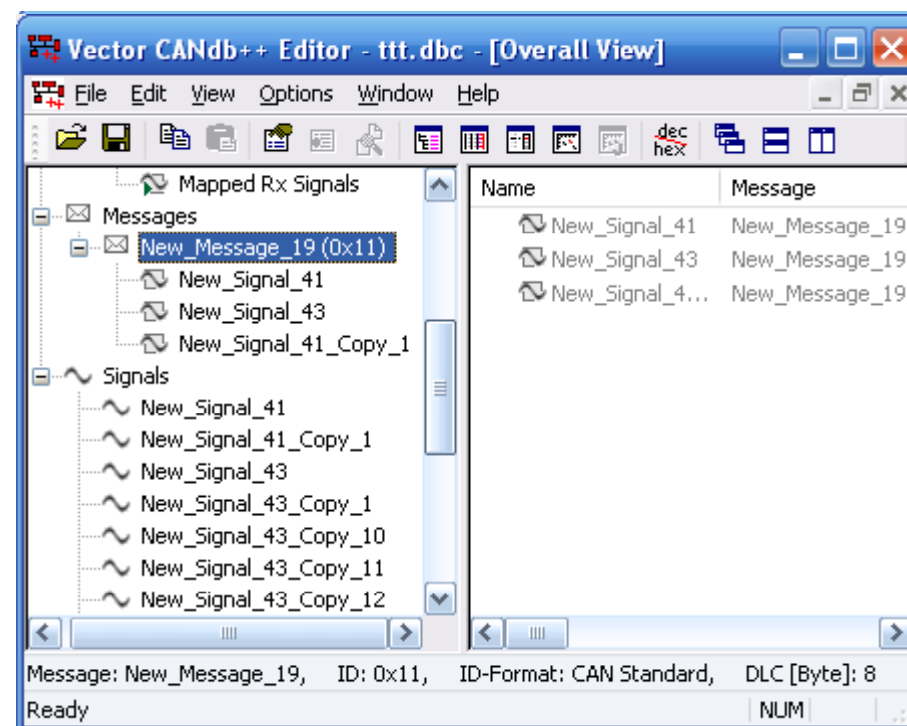
修改/编辑已有对象

▣ 直接双击



对象链接(1/2)

- 信号与报文之间的连接
- 发送报文与节点之间的连接
 - 鼠标拖拽或Copy-Insert



对象链接(2/2)

□ 接收报文与节点之间的连接

□ 通过信号间接定义

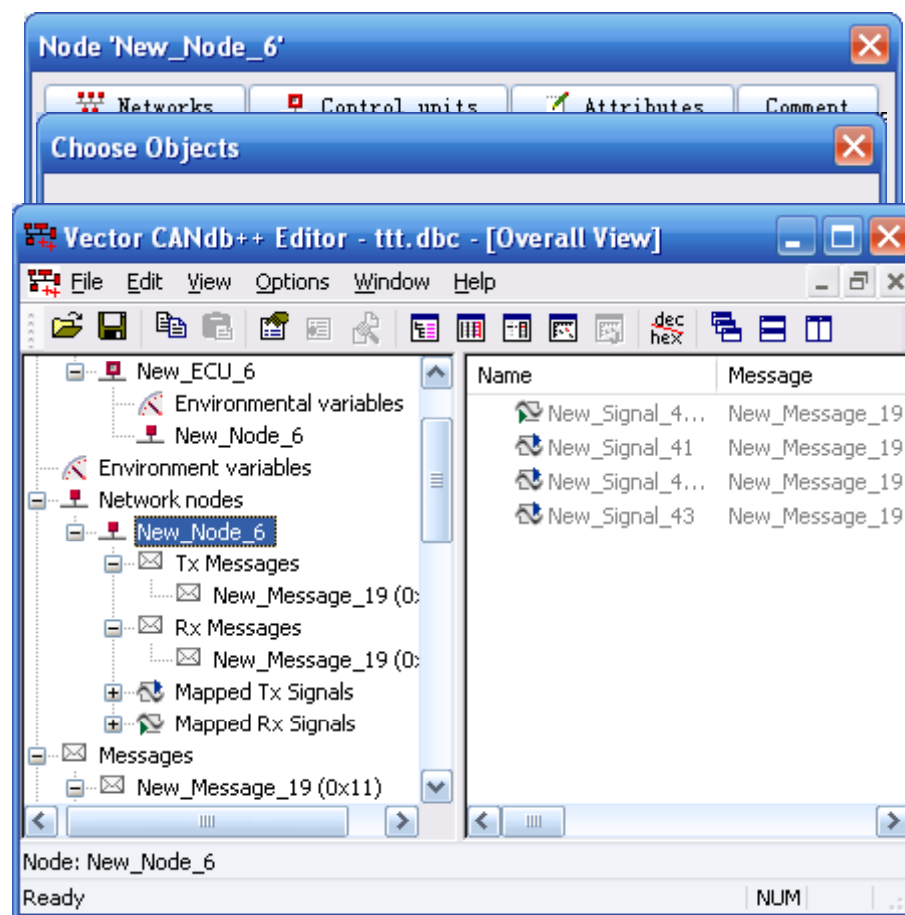
□ 双击节点，

选择Mapped Rx Sig.页签

□ 点击Add...，选择接收信号

点击OK

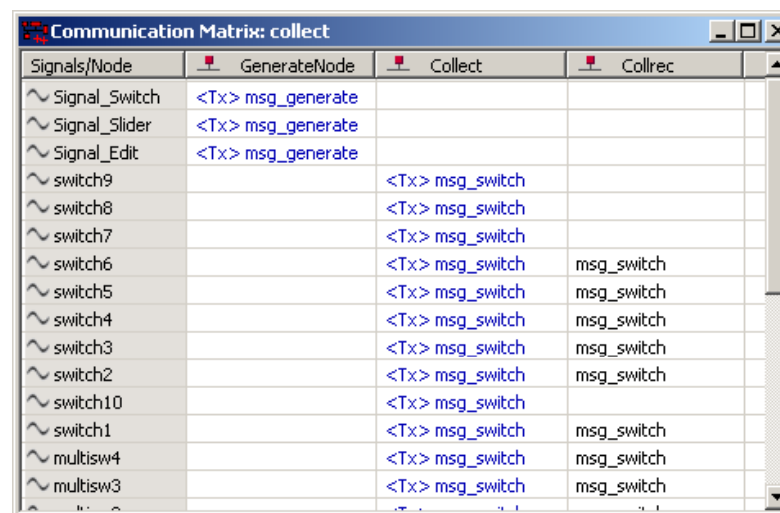
□ 点击确定



通信矩阵

□ View->Communication Matrix...

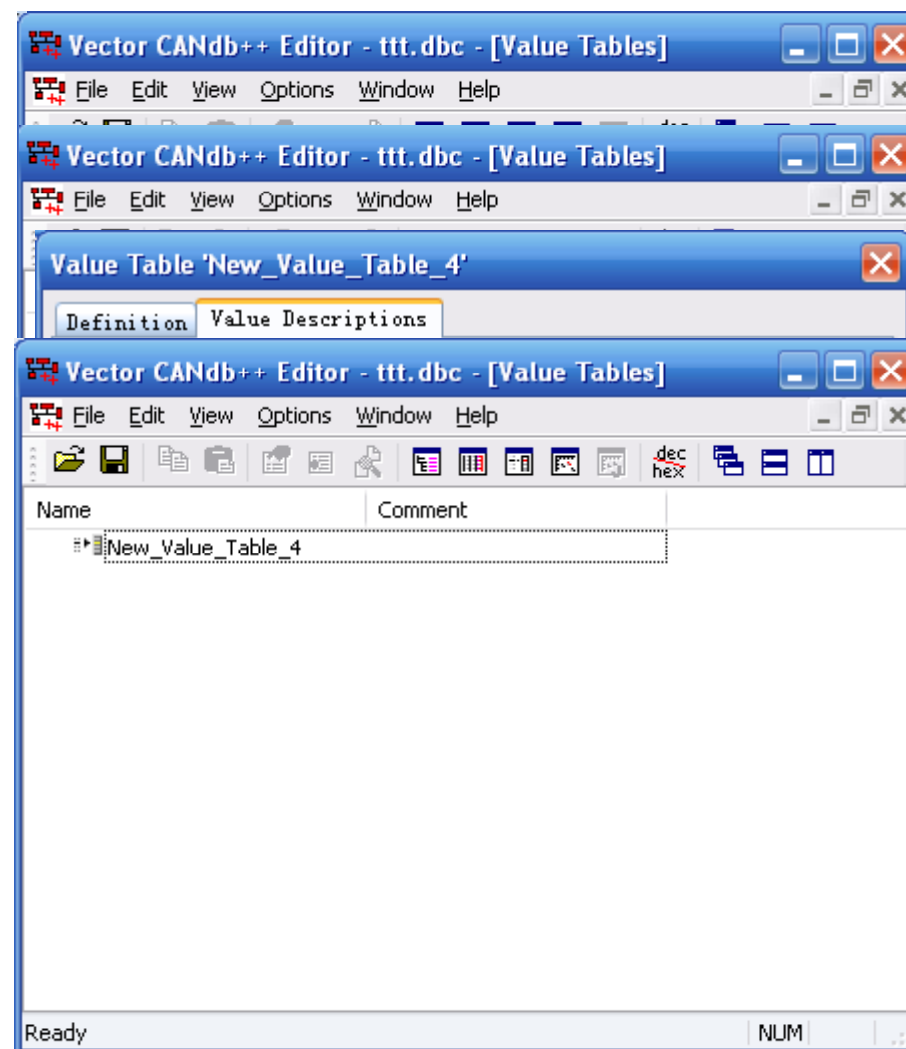
- 显示信号、消息、及网络节点的关系
- 以信号为行，网络节点为列
- 消息名显示于表中，对应了包含的信号与发送/接收的节点



Signals/Node	GenerateNode	Collect	Collrec
~ Signal_Switch	<Tx> msg_generate		
~ Signal_Slider	<Tx> msg_generate		
~ Signal_Edit	<Tx> msg_generate		
~ switch9		<Tx> msg_switch	
~ switch8		<Tx> msg_switch	
~ switch7		<Tx> msg_switch	
~ switch6		<Tx> msg_switch	msg_switch
~ switch5		<Tx> msg_switch	msg_switch
~ switch4		<Tx> msg_switch	msg_switch
~ switch3		<Tx> msg_switch	msg_switch
~ switch2		<Tx> msg_switch	msg_switch
~ switch10		<Tx> msg_switch	
~ switch1		<Tx> msg_switch	msg_switch
~ multism4		<Tx> msg_switch	msg_switch
~ multism3		<Tx> msg_switch	msg_switch

数值表(1/2)

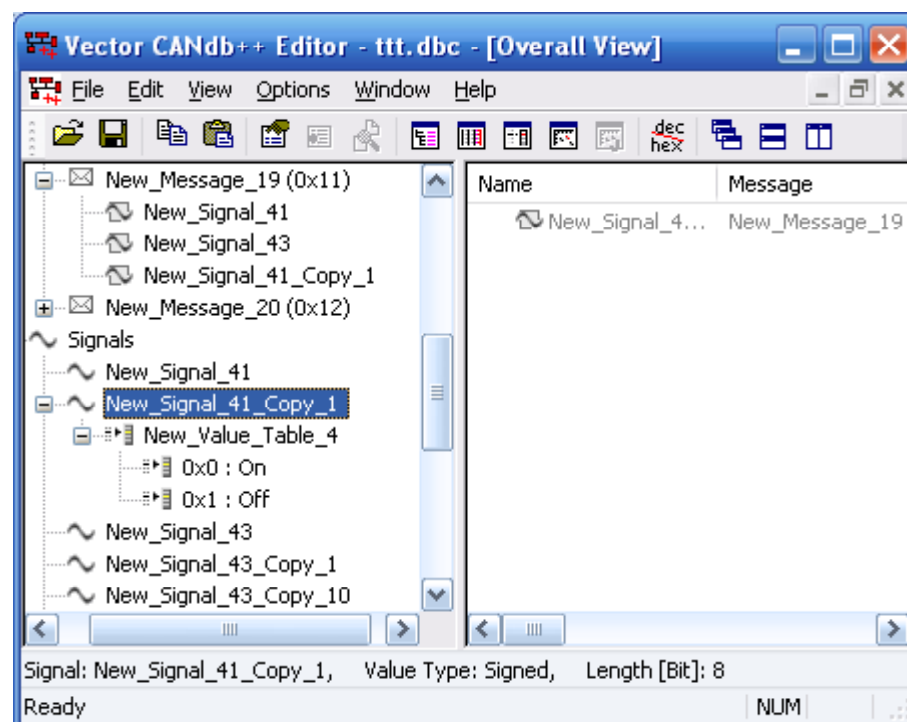
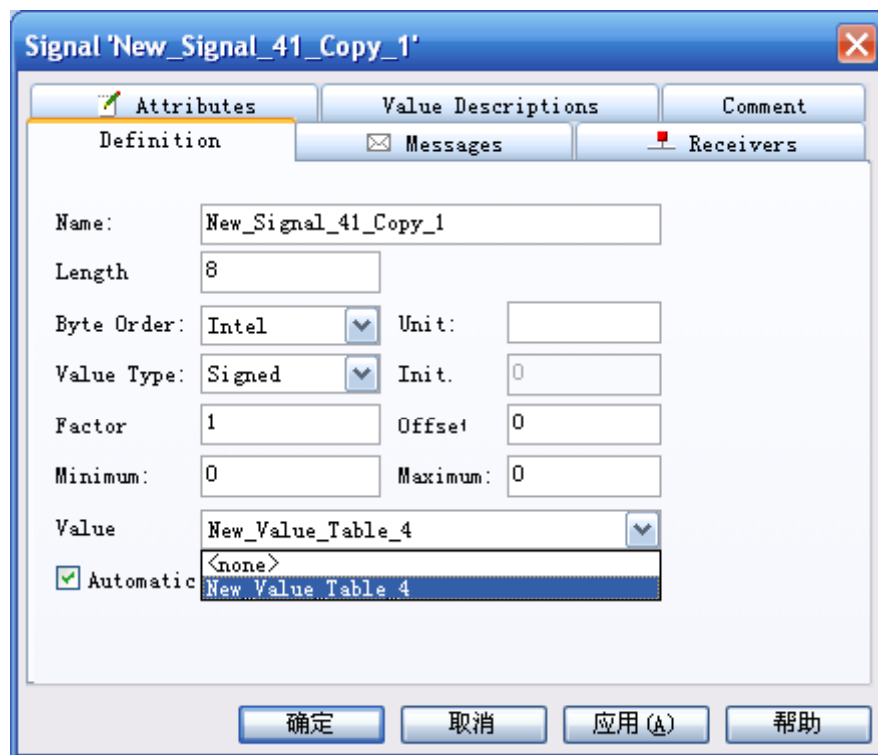
- 新建数值表
 - View->Value Tables
 - 右键点击空白处，
选择New...
 - 在对话框中输入数值，
点击确定
 - 新的数值表创建完成



数值表(2/2)

分配数值表

数值表可以分配给信号或环境变量



属性列表

□ Vector Tool Chain Attributes

□ General

□ Manufacturer

□ Interaction Layer

□ GenMsgCycleTime

□ Transport Protocol and Diagnostics

□ DiagRequest, DiaResponse

□ Network Management

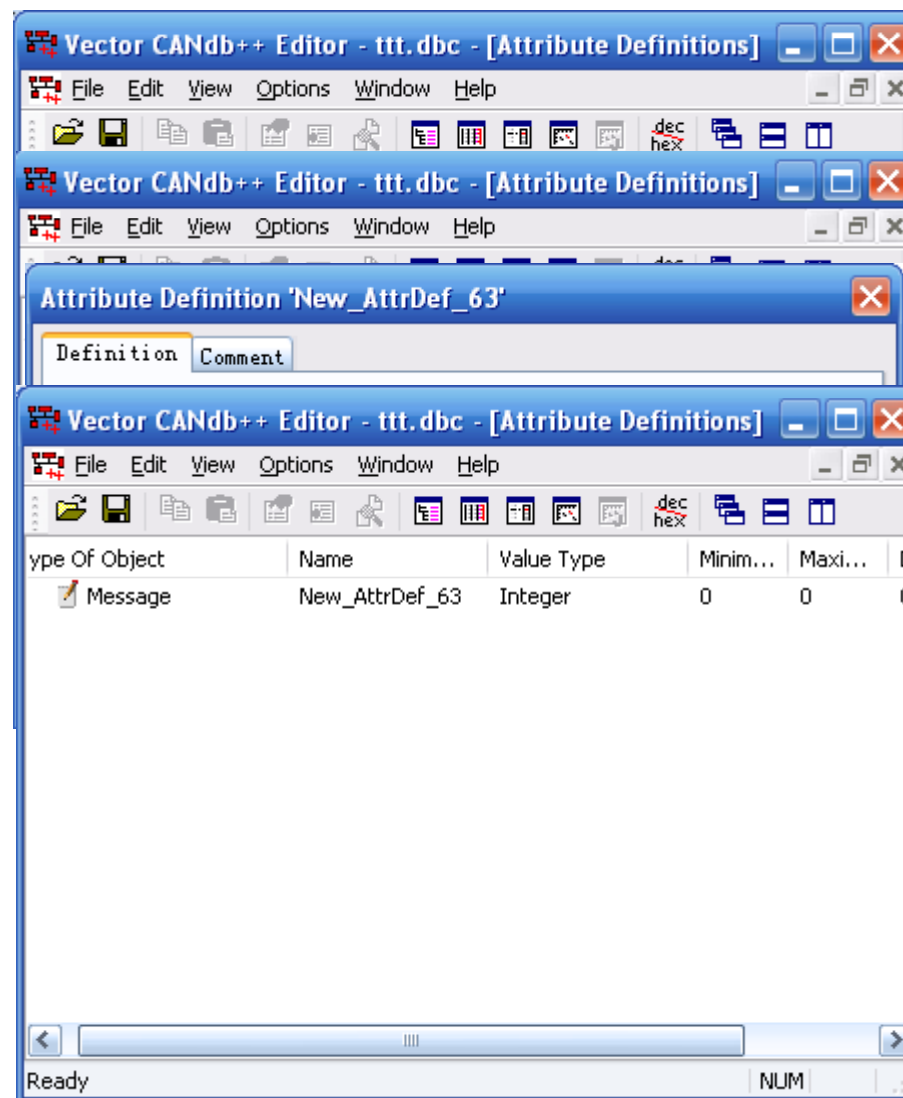
□ NmBaseAddress, NmStationAddress

□ Tool specific

□ BusType

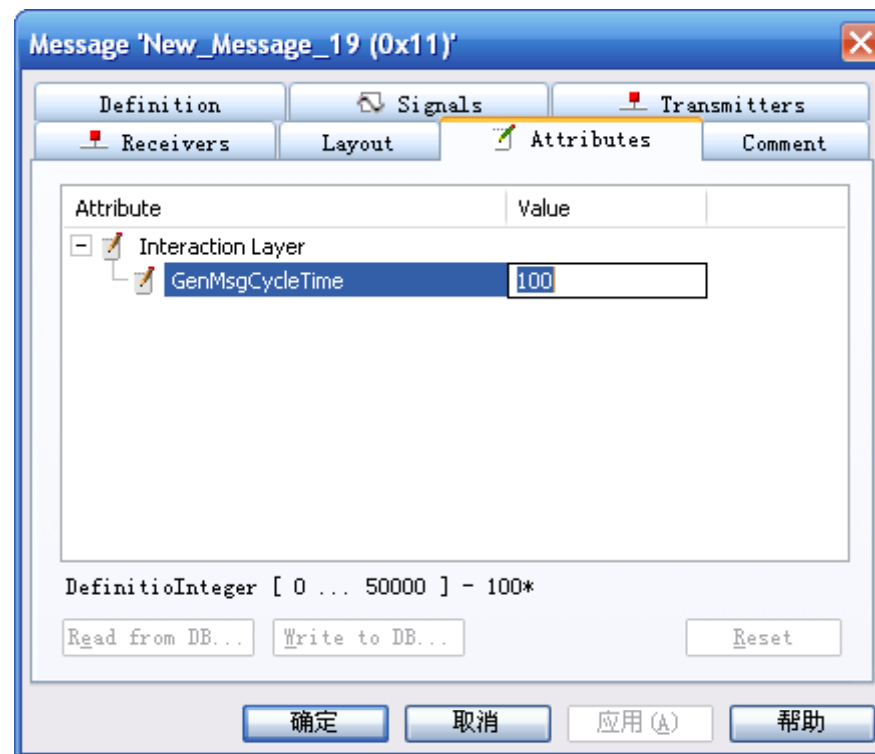
新建属性

- View->Attribute Definitions
- 右键点击空白处，
选择New...
- 在对话框中输入相关参数，
点击确定
- 新的属性创建完成



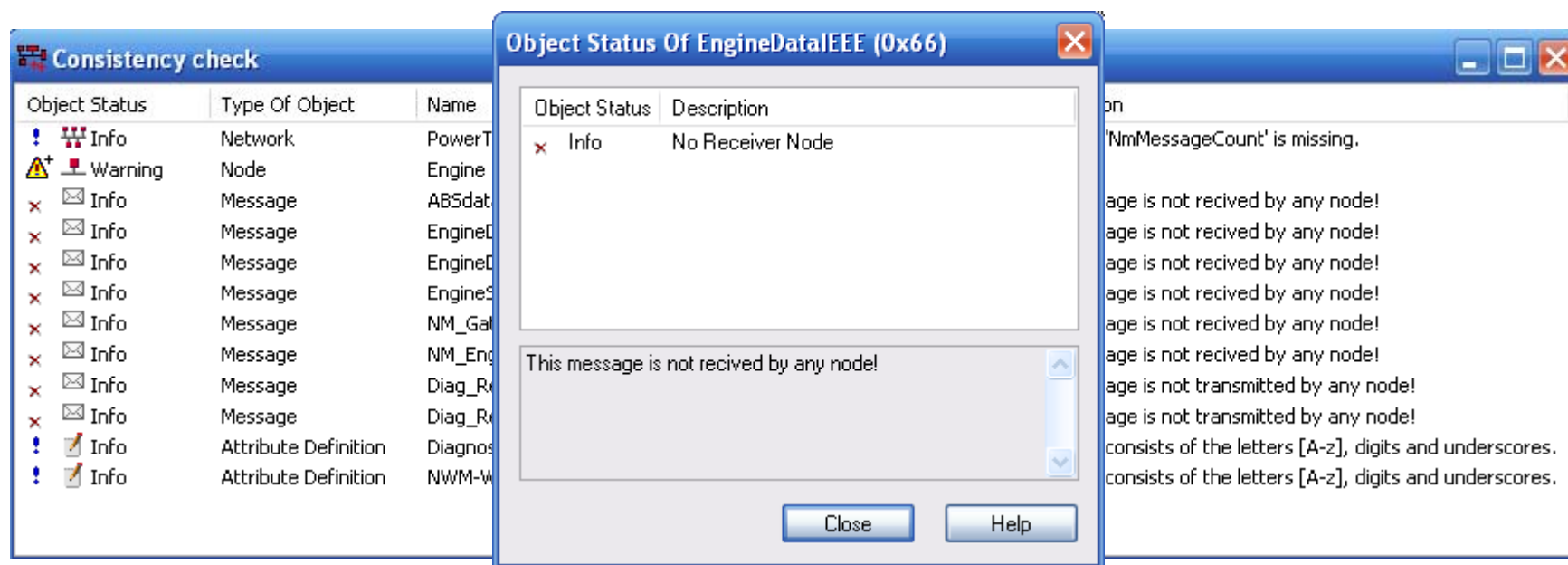
对象属性修改

▣ 双击对象



一致性检查

□ File-> Consistency Check

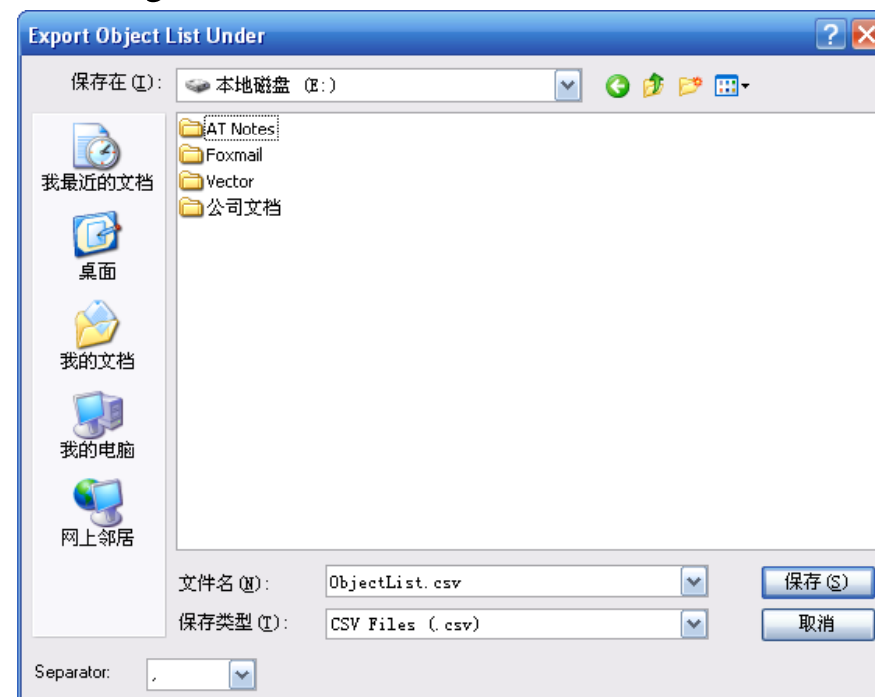


数据导出

□ 选择需要导出的对象

□ 信号，报文，节点，ECU或网络

□ File->Export->Export List of Objects



创建面板/虚拟仪表

□ Panel Editor

- 传统的面板编辑器

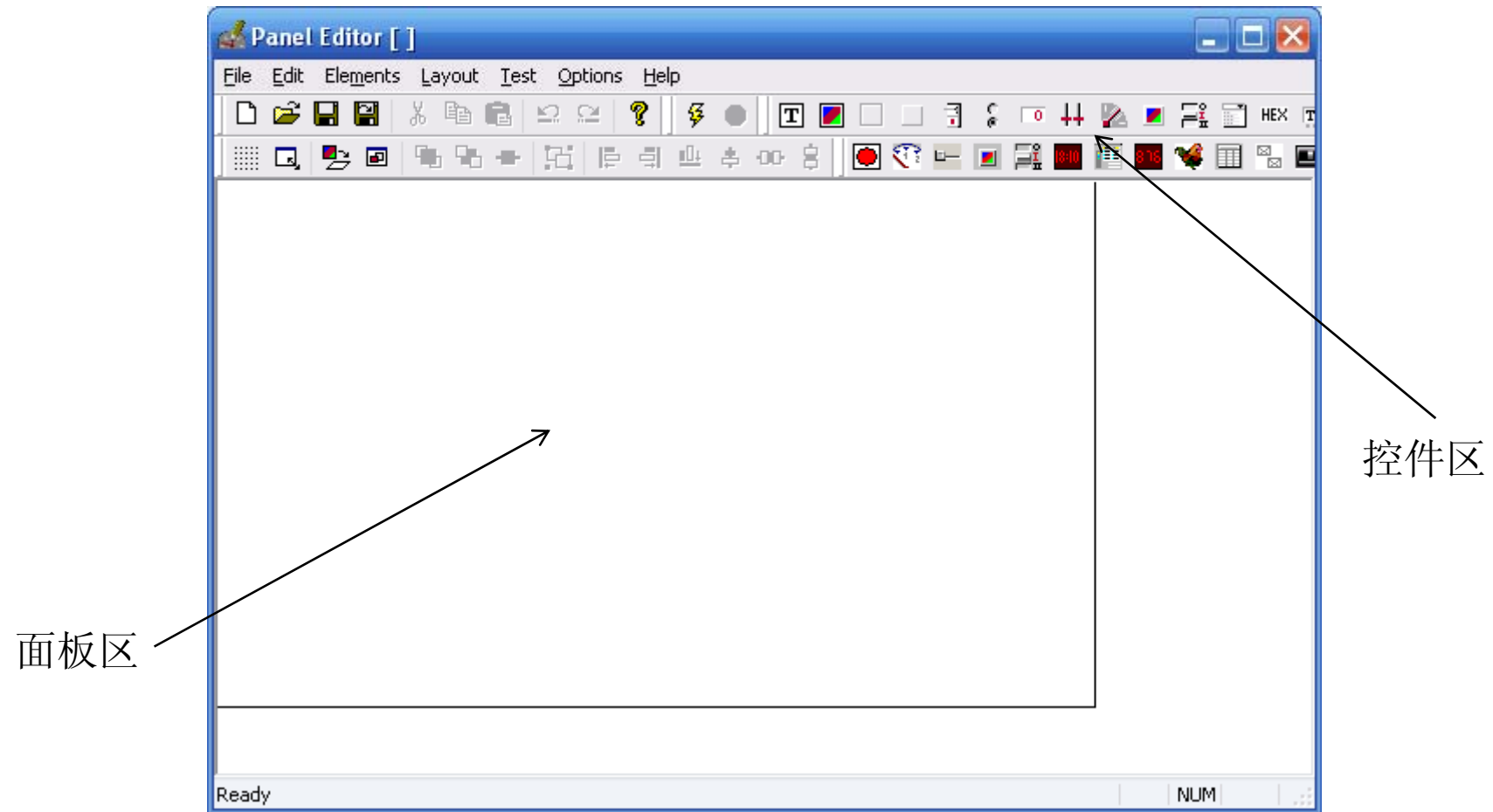
- File->Open Panel Editor

□ Panel Designer

- 新的面板编辑器

- File->Open Panel Designer

Panel Editor



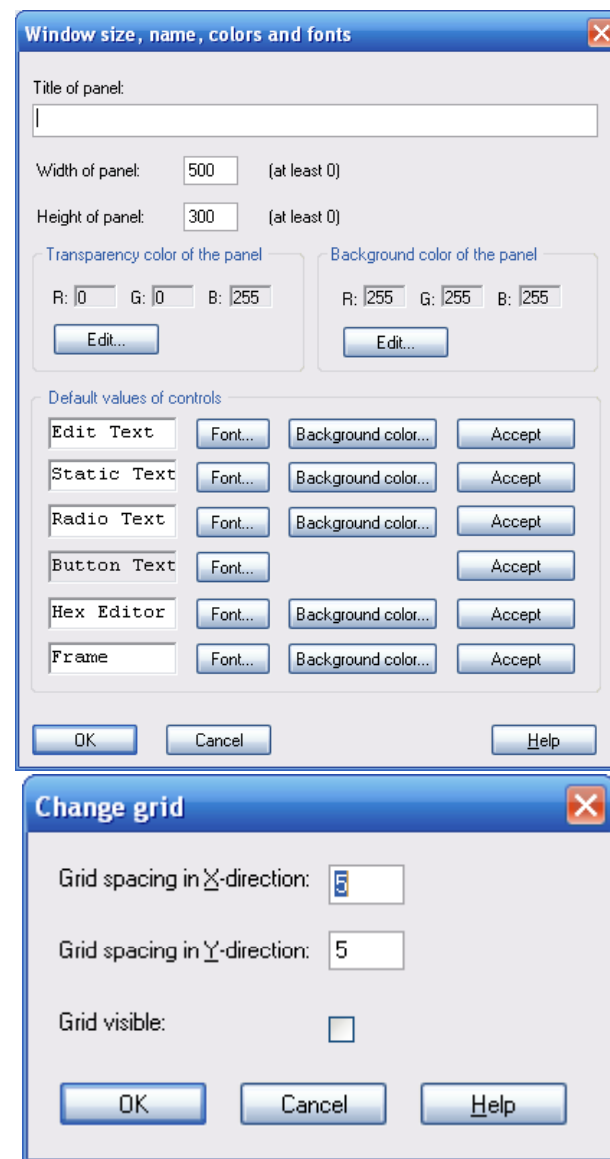
面板设置

□ Options->Window setting

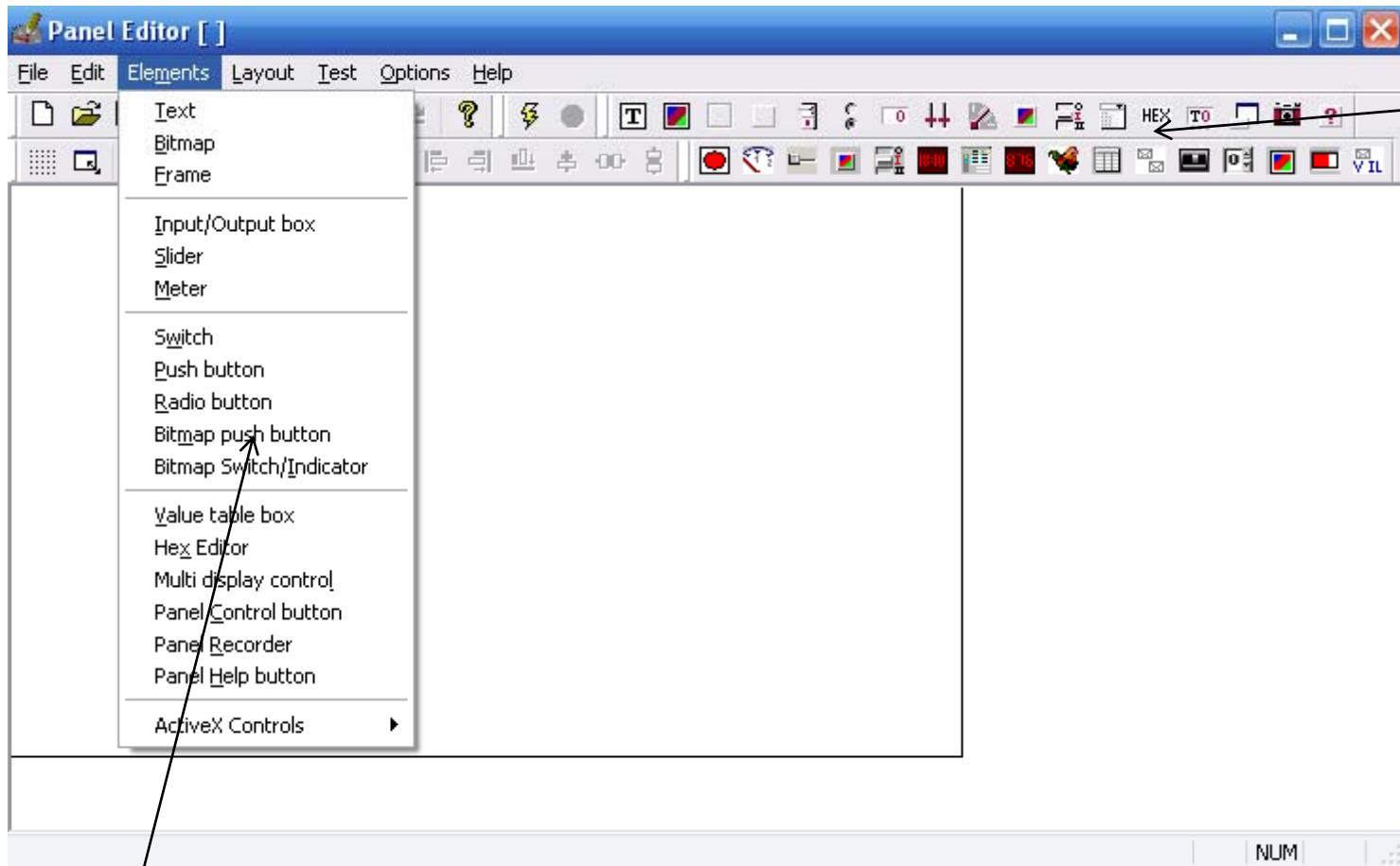
- 定义面板名称
- 面板尺寸
- 背景颜色
- 透明色
- 控件的缺省字体和颜色

□ Options->Change grid

- 网格大小
- 网格可视



控件列表

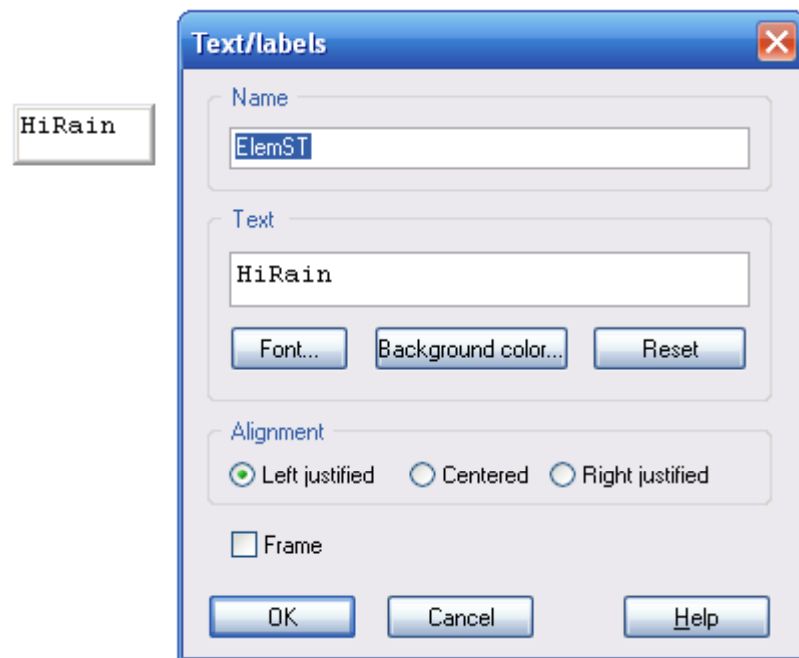


图标

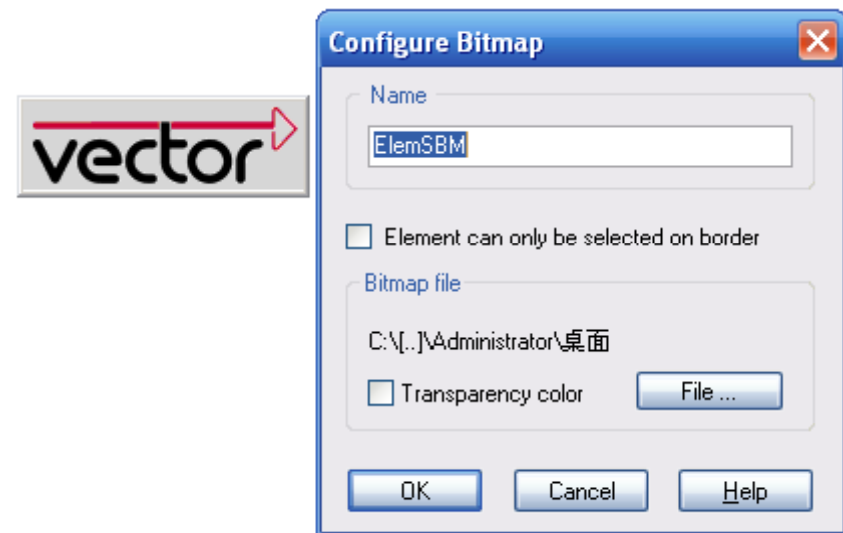
名称

文本与位图

□ 文本

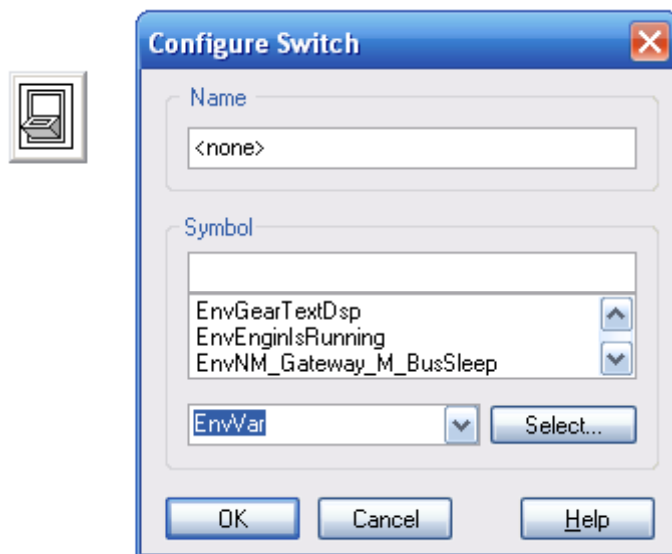


□ 位图

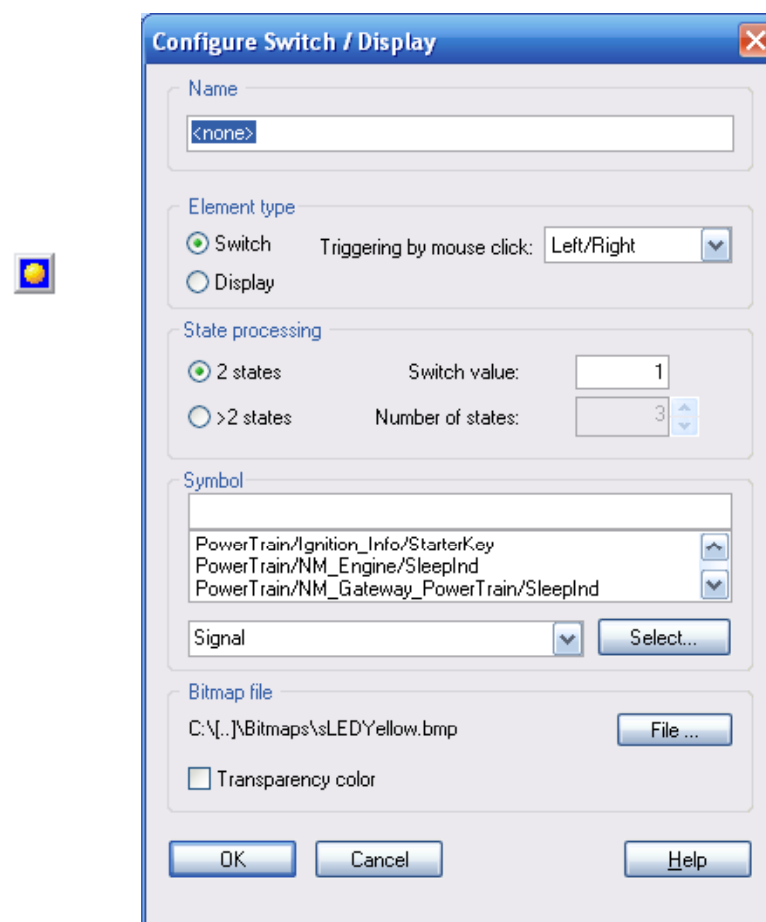


开关与多态位图

□ 开关

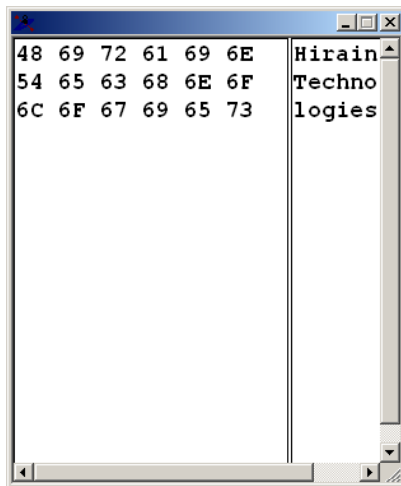


□ 多态位图



其它常见的控件

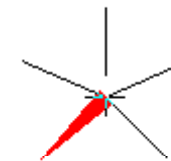
□十六进制



□滑动条



□仪表



□输入/输出显示



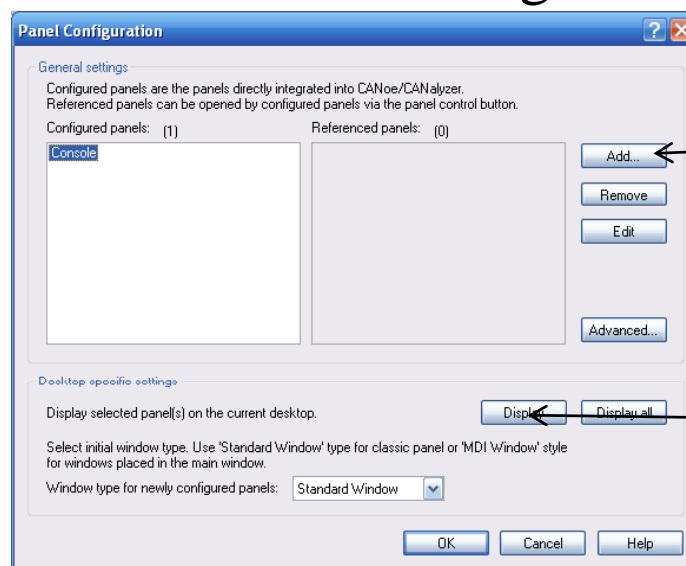
面板使用

□ 保存面板

□ File->Save

□ 使用面板（CANoe）

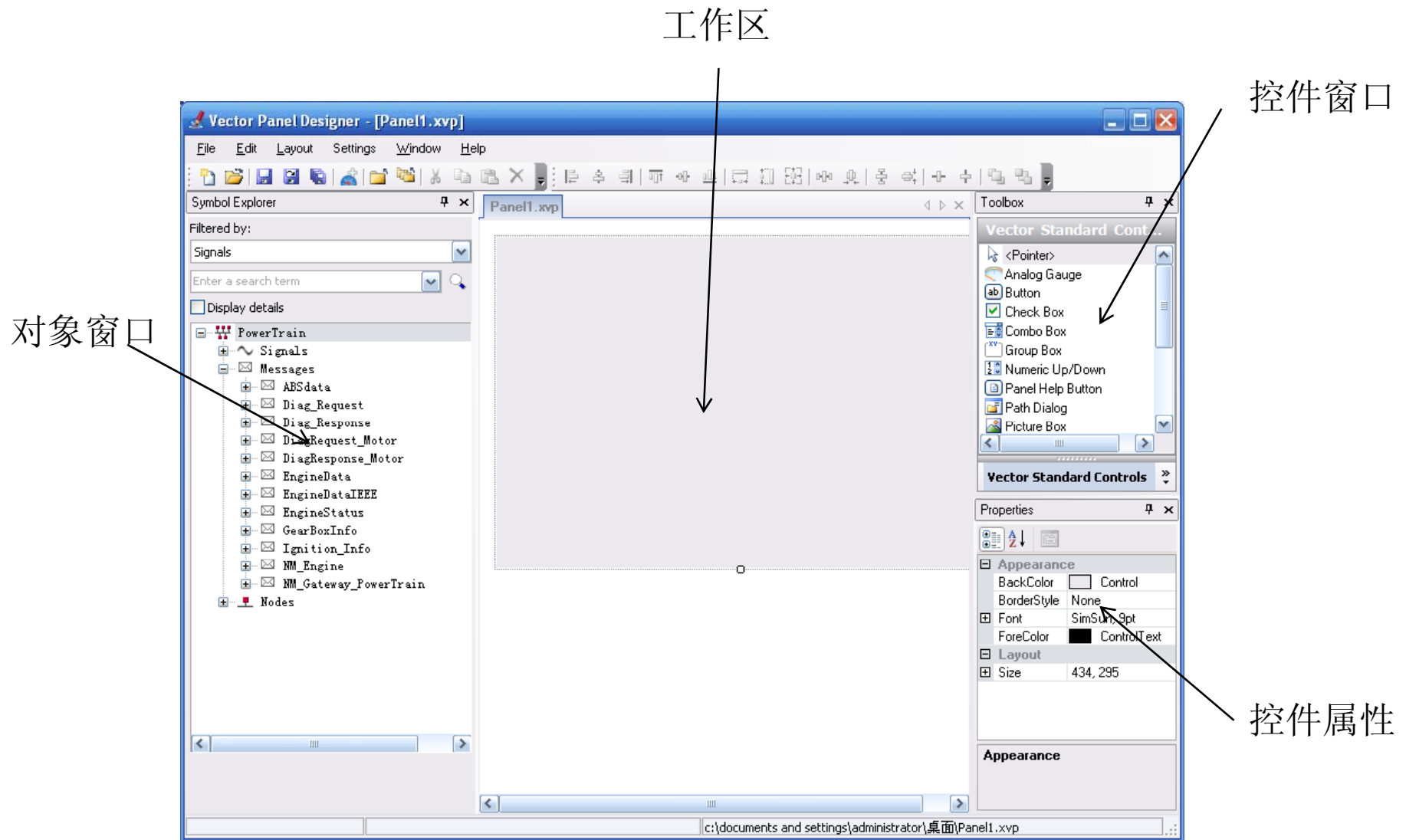
□ Configuration->Panel Configuration



添加面板

显示面板

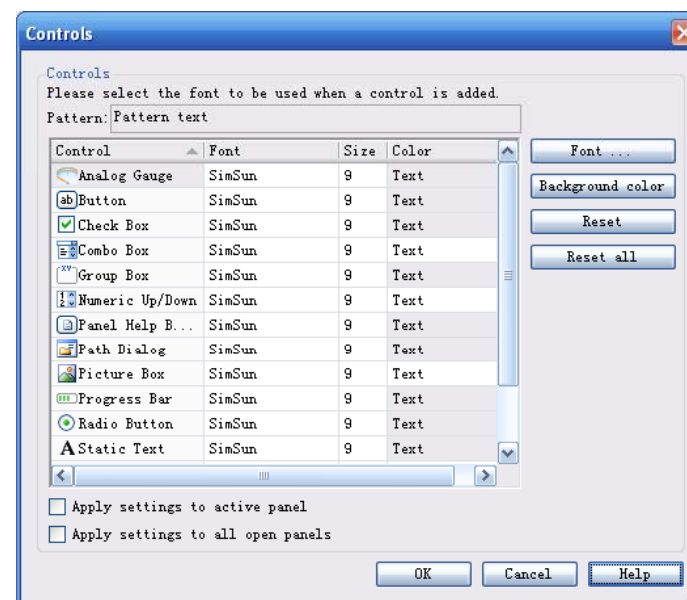
Panel Designer



设置

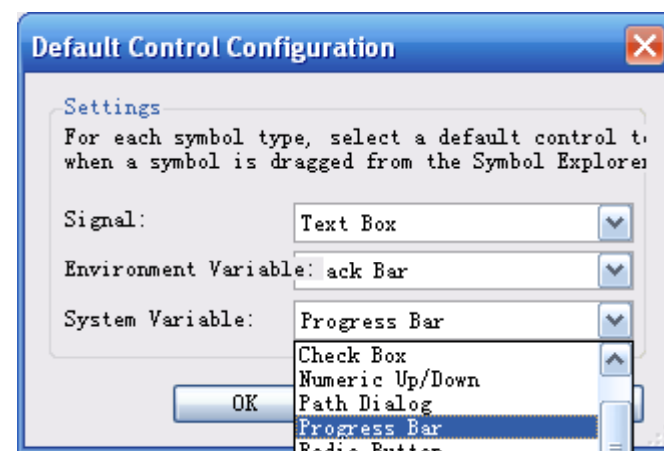
□ Settings->Controls Properties

□ 设置控件的字体、颜色和字号



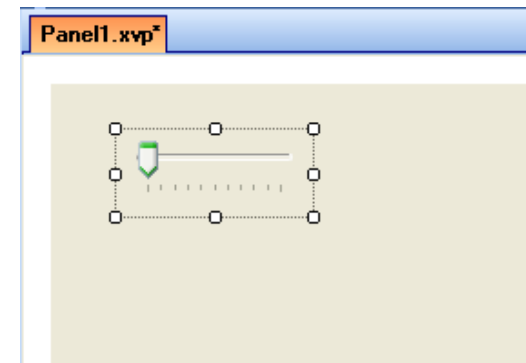
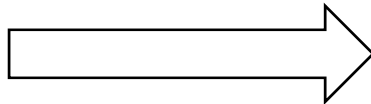
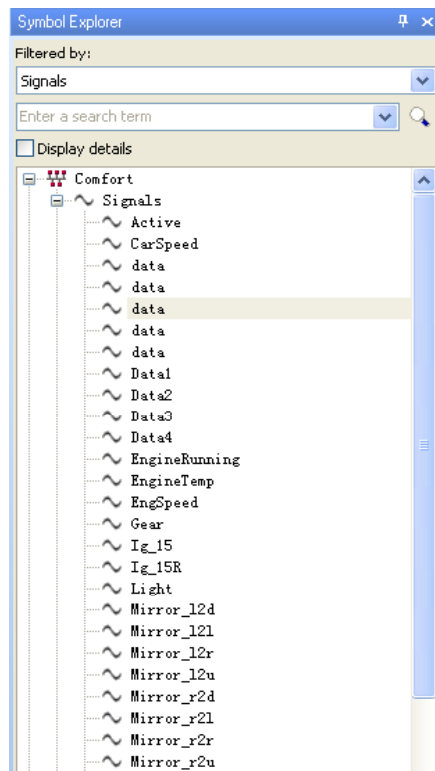
□ Settings->Symbol Explorer

□ 设置信号、环境变量和系统变量
对应的默认控件



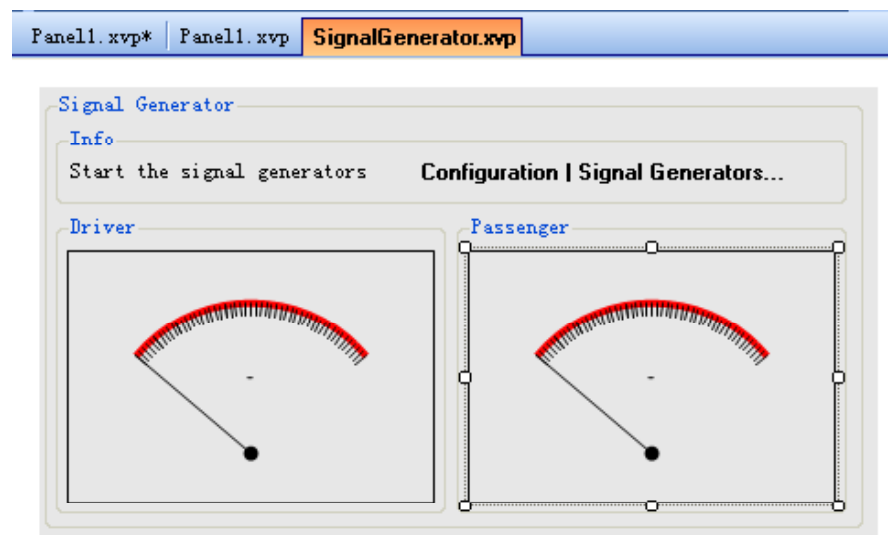
对象窗口

- ▣ 显示信号、环境变量和系统变量
- ▣ 直接拖拽变量到工作区生成控件



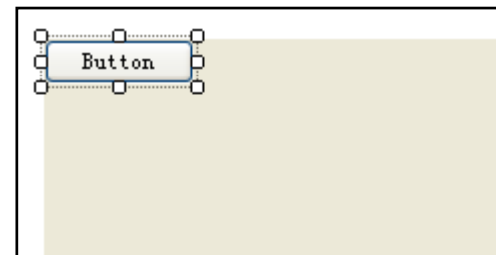
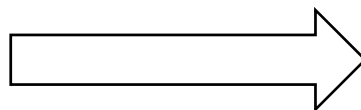
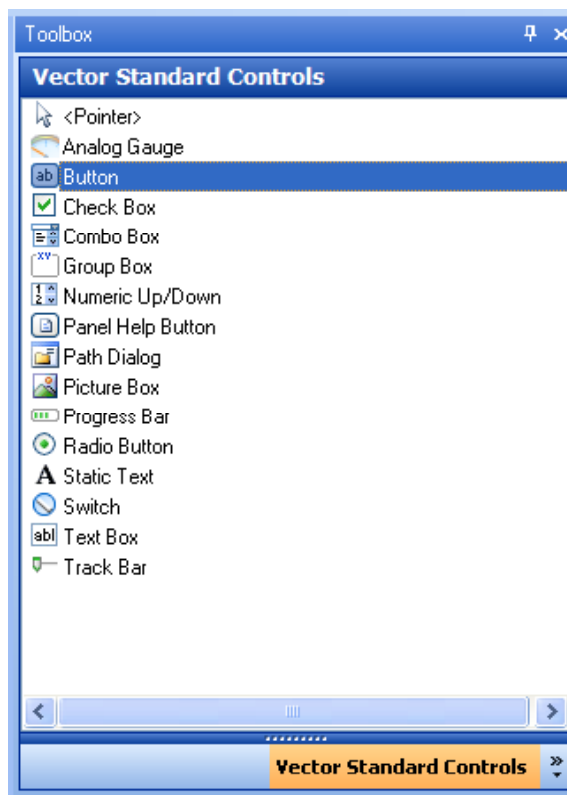
工作区

- 创建面板
- 支持同时编辑多个面板



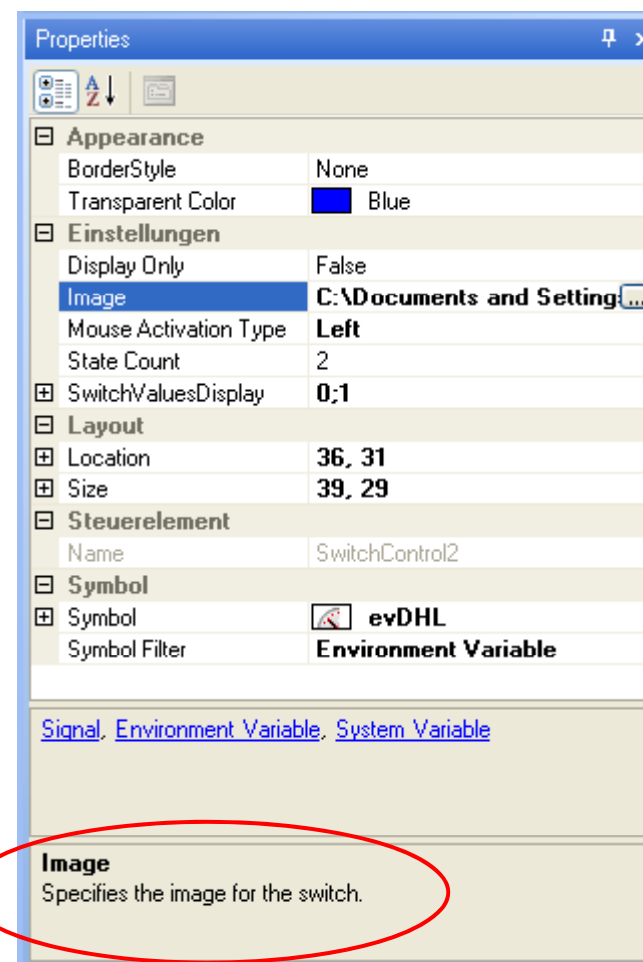
控件窗口

- 显示控件
- 双击在工作区产生控件

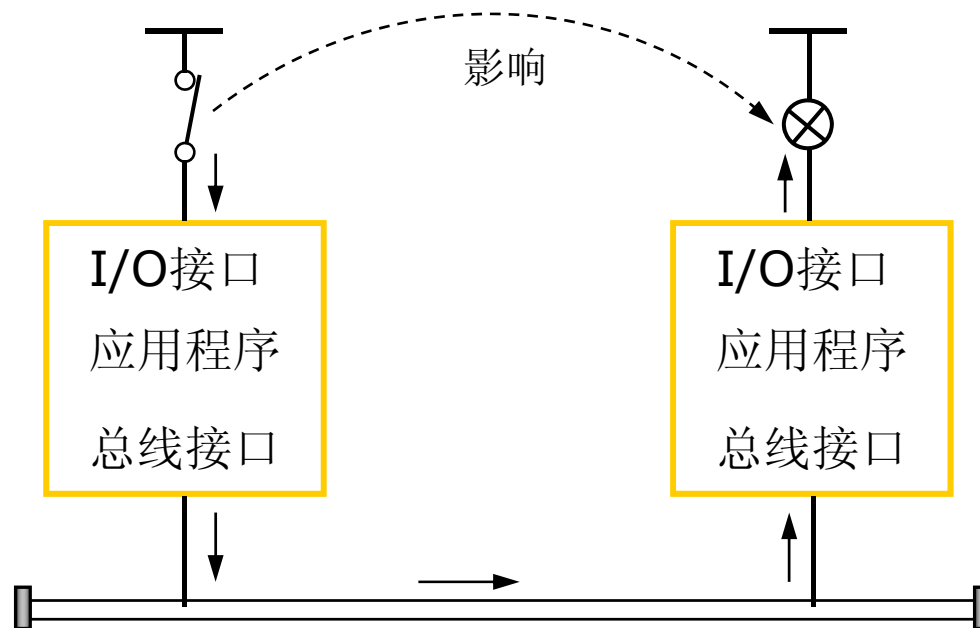


控件属性窗口

- 显示选中控件的相关设置
- 点击某项设置后会在下方出现相关说明



练习1



练习2



小提示

- ▣ Measurement Setup窗口和Simulation Setup窗口是CANoe的主要窗口，进行数据流规划
- ▣ 几乎窗口中的所有对象均可通过点击鼠标右键来访问交互菜单
- ▣ 所有数据传输到评估模块时，均会在对应窗口以各自的方式进行显示，记录模块除外
- ▣ 配置文件可以保存CANoe中的所有设置；可以使用已有的配置文件作为新任务的基础，进行简单的修改形成新的配置，提高效率