在 WinCC 中，如果某个变量组态在系统中出现多次，可以使用结构类型。 例如，根据同一原理设置的多个电机，那么可以创建一个名为 Motor 的结构类型，其中的每个变量都由一个结构元素来表示，例如 ON_OFF、SetValue、ActualValue 等。每次使用该结构类型创建新结构实例时，WinCC 都会为相应的电机自动生成所有结构变量。

例如，当对希望通过画面窗口来集成的画面进行组态时，可使用所创建的结构变量。创建一个对应电机的画面模板，然后在画面中多次调用画面窗口，连接该模板但关联不同的结构变量，能够实现在各个窗口中显示各个电机不同的状态。

本文描述的是利用结构变量和画面窗口共同组态实现画面模板的两种方法。

1，使用变量前缀的画面窗口

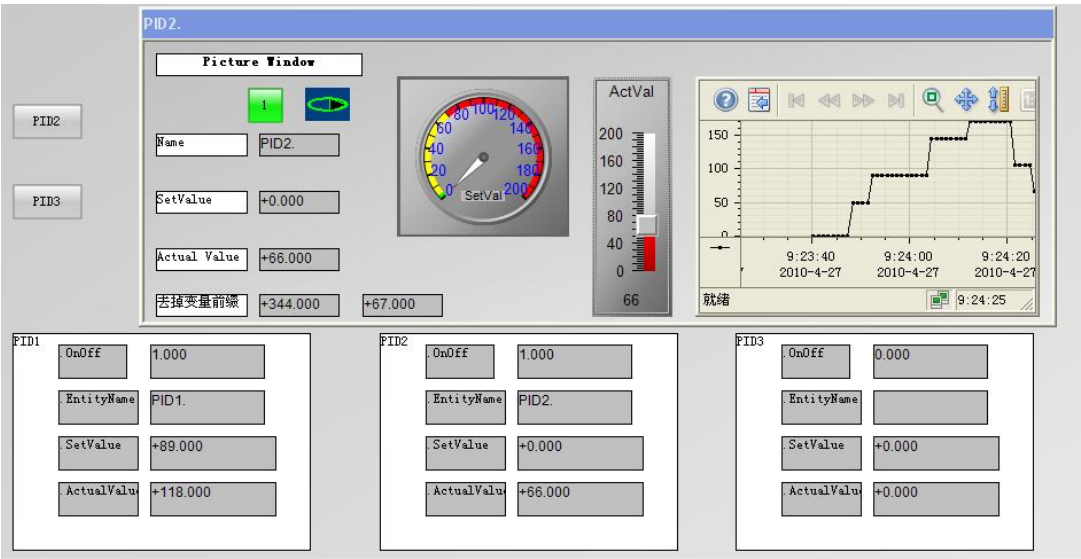

图 1，画面窗口的样例

首先组态一个模板画面，画面中的对象不是与变量相连，而是与结构元素相连。在运行系统中，WinCC 通过画面窗口的变量前缀以及已链接在模板画面中的结构元素的名称来构成所需结构变量的名称。

“变量前缀”属性将指定画面中出现的所有变量的前缀。前缀可自由定义，但必须与结构变量的名称相匹配，它必须以句点结尾，例如“Structure2.”。改变变量前缀只有在再次装载画面时才起作用。使用名称 TagPrefix 可使“变量前缀”属性动态化；涉及到控件时仅为“WinCC 量表控件”和“WinCC 滚动条控件”提供 TagPrefix 属性。

1.1 组态一个作为模板的画面

1.1.1 创建结构变量

在项目中创建一个名为 PID 的结构类型，包含四个结构元素。

图 2，创建结构类型

然后，创建三个结构实例 PID1，PID2，PID3，分别对应现场的三台电机，WinCC 自动生成结构变量。

1.1.2 创建画面模板

创建一个画面 PicModule.pdl，画面中的对象包括 IO 域，按钮，状态显示，量表控件，滚动条控件，趋势图等，画面中的对象连接到结构元素上。

1.1.2.1 IO 域的组态

在画面中添加 IO 域，为每个 IO 域连接变量，如 PID1.EntityName。

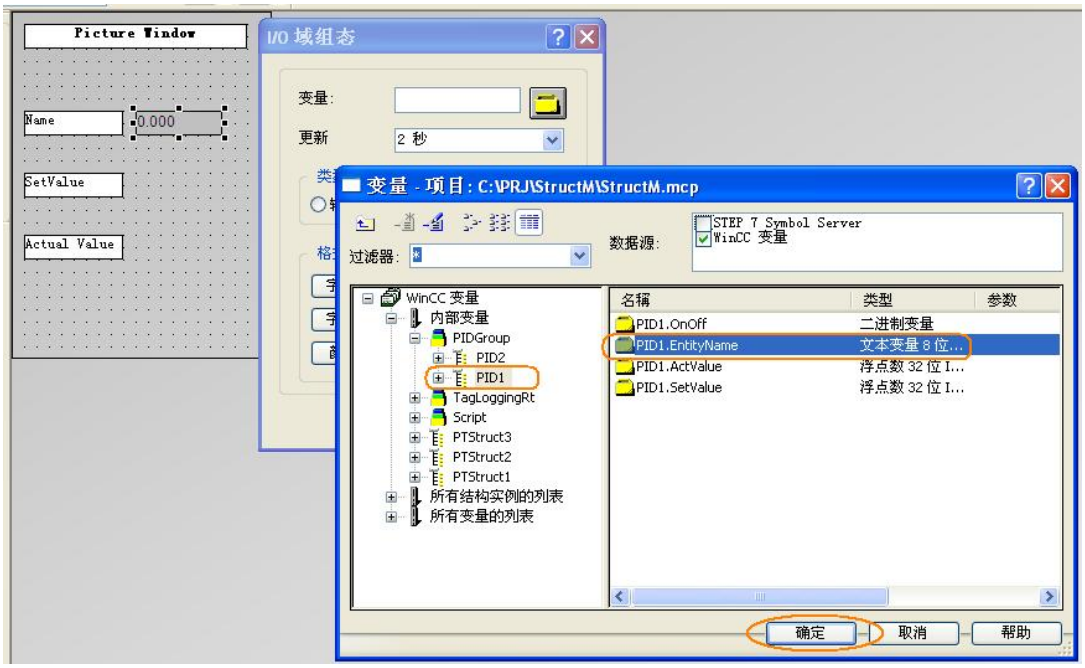

图 3，组态 IO 域

删除变量的前缀，仅保留结构元素部分，如 EntityName，如图 4 所示。

图 4，删除变量前缀

其他 IO 域的组态方法类似，分别连接 EntityName、ActValue、SetValue 三个元素，用来显示电机名称，电机实际转速和电机设定转速。

1.1.2.2 按钮的组态

再添加一个按钮，用来控制电机启停，并根据启停状态改变颜色，显示不同的文本。

在按钮的事件中组态 C 动作，C 动作中的变量参数输入结构元素”OnOff”，代码如下：

```
#include "apdefap.h"
void OnClick(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName)
{
#pragma option(mbcs)
BOOL a;
a=GetTagBit("OnOff");      //Return-Type: BOOL
SetTagBit("OnOff",1-a);    //Return-Type: BOOL
}
```

为了使按钮的颜色动态更改，设置背景颜色的属性，组态动态对话框，在“表达式”一栏选择变量，如图 5。

图 5，按钮颜色动态的组态—连接变量

同样需要删除变量前缀，仅保留元素部分，组态后按钮的背景颜色会根据”OnOff”的不同数值而切换。

图 6，按钮颜色动态的组态—删除前缀

如上图所示的组态完成后，点击“应用”后，系统会因为找不到变量”OnOff”而报图 7 的警告。

图 7，警告

这时可以选择创建一个同名变量，也可以“忽略”这个警告。

需要注意的是，对于 WinCC V7，所有的对象默认都是应用“全局颜色方案”的，这样的对象颜色无法动态更改，因此需要设置对象的“全局颜色方案”属性为“否”，如图 8。

图 8，修改全局颜色方案

按钮的文本组态为由变量动态设置，显示的是所连接变量的数值，如图 9。

图 9，按钮文本的动态组态

按钮的组态已完成，该按钮能够控制电机启动/停止，并根据电机的启停状态显示不同的文本和颜色。

1.1.2.3 状态显示的组态

接着组态一个状态显示对象，根据电机的启停显示不同的图形，状态显示的组态界面如图 10，同样使用的是结构元素”OnOff”。

图 10，状态显示的组态

1.1.2.4 量表和滚动条的组态

由于只为“WinCC 量表控件”和“WinCC 滚动条控件”提供了 TagPrefix 属性，因此在画面中添加一个量表控件和一个滚动条控件。

量表控件的组态如图 11，控件的“数值”属性连接结构元素”SetValue”，更新周期为“有变化时”。

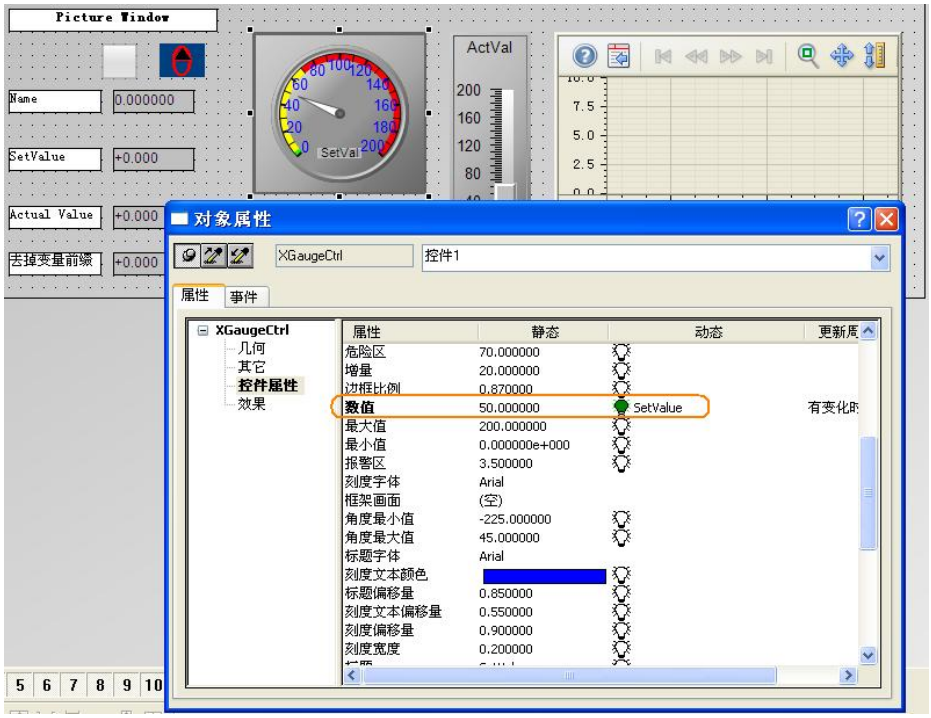

图 11，量表控件的组态

滚动条控件的组态如图 12，滚动条的“位置”属性连接结构元素”ActValue”，更新周期是“有变化时”。

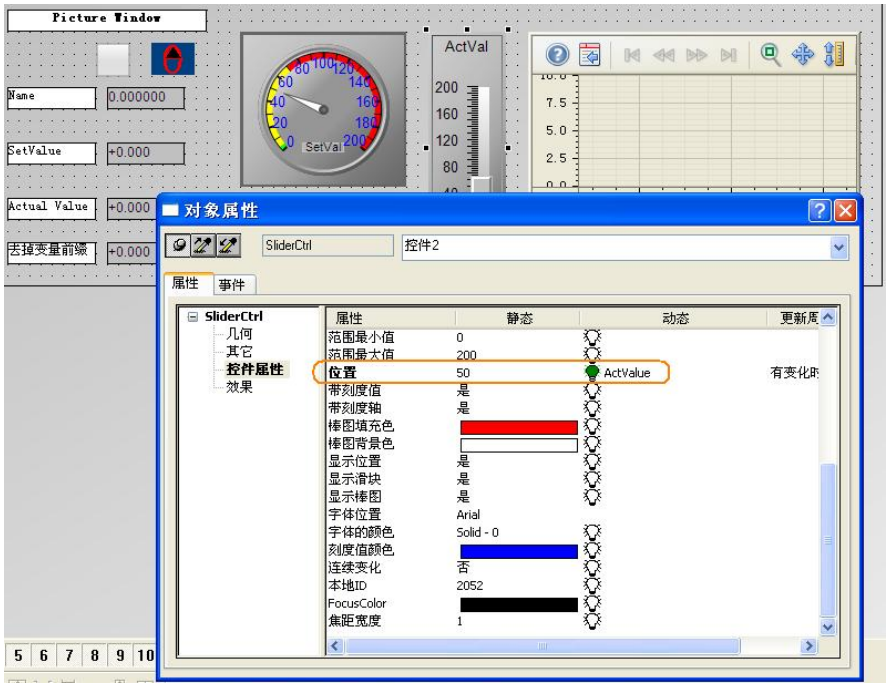

图 12，滚动条控件的组态

为了通过滚动滑块的位置变化控制变量值，为滚动条的事件“位置—更改”组态直接连接，直接连接的源是滚动条的“位置”属性，目标是结构元素”ActValue”，如图 13。

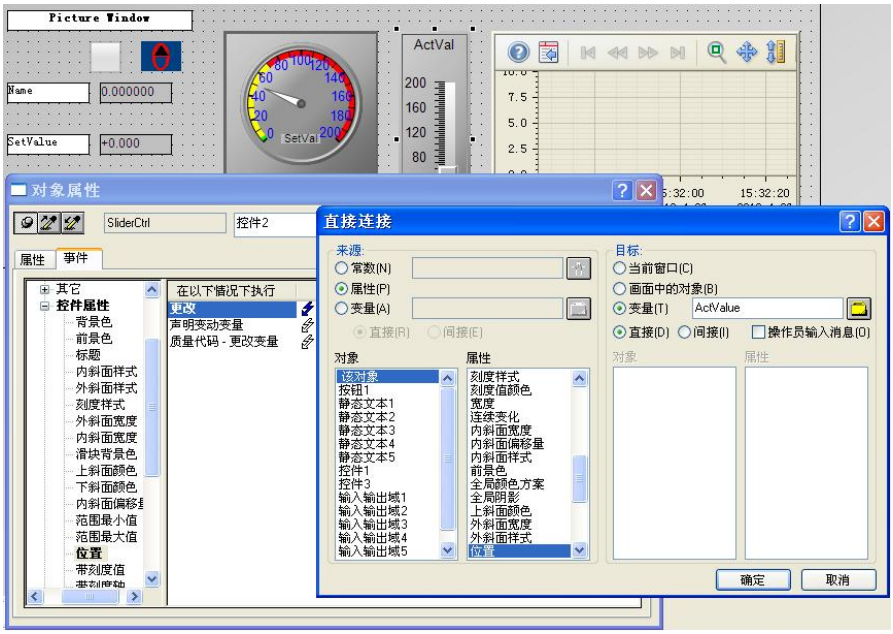

图 13，滚动条控件的事件组态

1.1.2.5 趋势控件的组态

以上是以某些对象为例说明了带有 TagPrefix 属性的对象如何关联到结构元素上，那么对于没有 TagPrefix 属性的对象，该如何处理呢？

以趋势控件为例说明。

（1）如果趋势控件关联“在线变量”。趋势图关联的变量无法使用画面窗口的 TagPrefix，因此仅在 TrendTagName 中连接结构元素是无法和 TagPrefix 一起组成结构变量的，处理方法是在控件属性”TrendTagName”中添加一个 C 动作，如图 14。

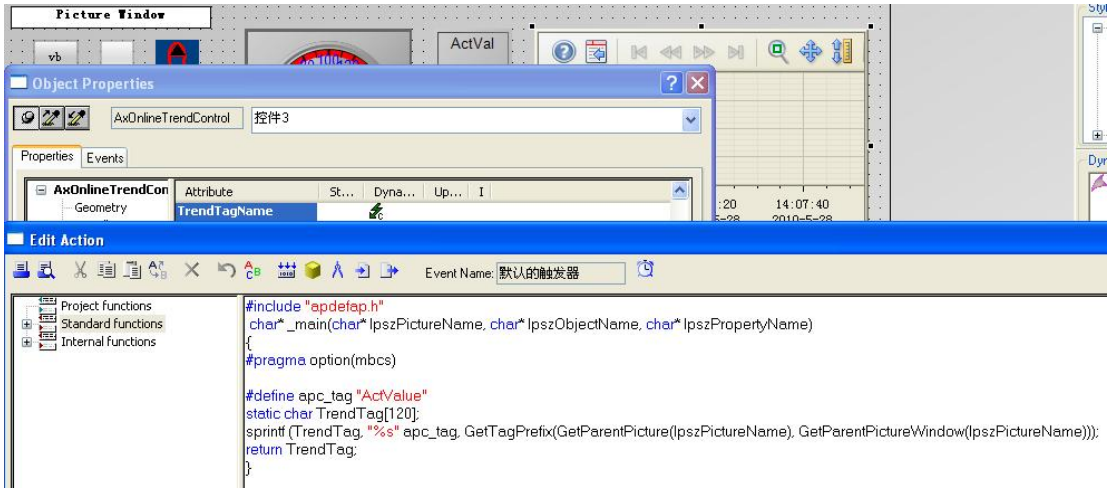

图 14，趋势控件的组态---在线变量

（本段代码由网友“玄极道人”提供）

代码 GetTagPrefix (GetParentPicture(lpszPictureName), GetParentPictureWindow (lpszPictureName)) 获得调用此模板的画面窗口的 TagPrefix，那么 sprintf 将 TagPrefix 和结构元素”ActValue”组合到一起，成为结构变量。

（2）如果趋势控件关联“归档变量”。假设归档名称为 ProcessValueArchive，那么代码需要改为 sprintf (TrendTag, "ProcessValueArchive\\%s" apc_tag, GetTagPrefix (GetParentPicture(lpszPictureName), GetParentPictureWindow(lpszPictureName)));，如图 15。

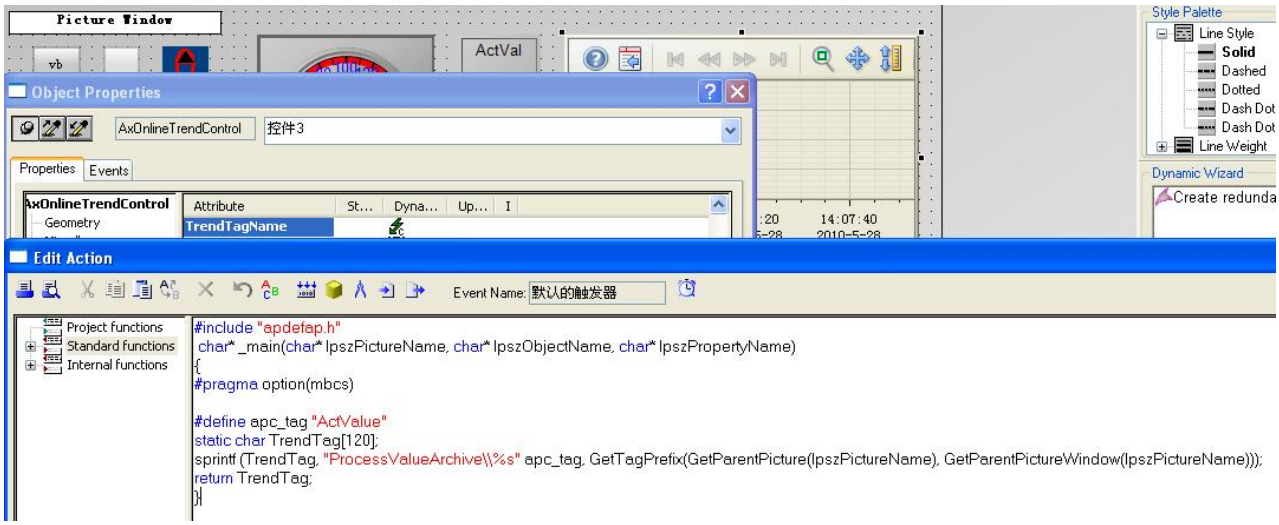

图 15，趋势控件的组态---归档变量

（本段代码由网友“玄极道人”提供）

至此，我们就组态好了一个作为模板的画面。

说明：为了更好的举例说明，本文组态的画面中调用了一些有特点的对象，用户可根据自己的需要酌情组态，请参考文中所描述的方法，但本文所述方法并非唯一，用户可自行修改。

1.2 创建一个新画面，调用模板

新建一个画面，调用多个画面窗口，通过设置不同的变量前缀，实现每个窗口显示一个电机参数值的效果。

1.2.1 静态设置 TagPrefix

直接设置画面窗口的“变量前缀”属性静态值为”PID1.”，那么这个画面窗口中的所有变量都添加了一个前缀 PID1.，成为结构变量”PID1.***”，组态方法如图 16。

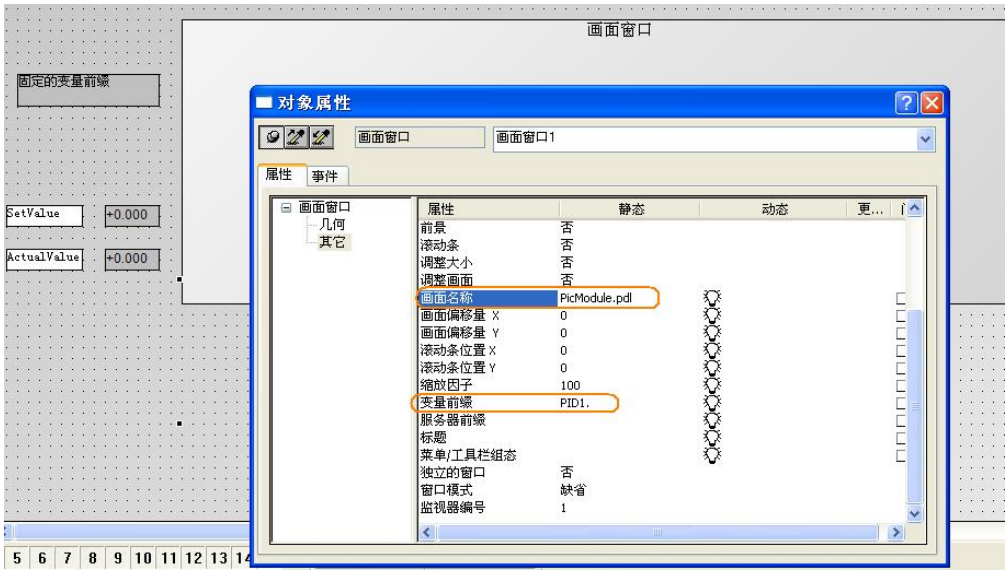

图 16，设置固定的变量前缀

运行后的效果如图 17。

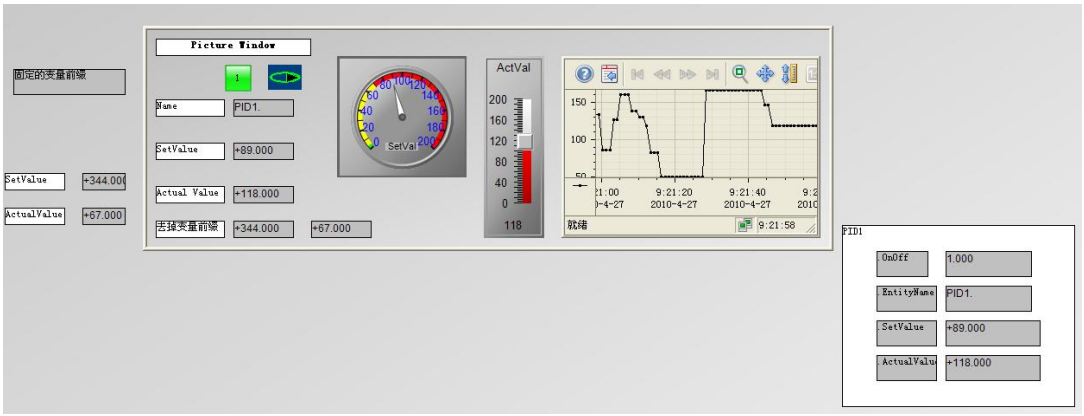

图 17，运行效果

1.2.2 通过变量修改 TagPrefix

设置画面窗口的“变量前缀”属性为一个变量 Prefix，如图 18，运行后由 Prefix 变量动态更改画面窗口的变量前缀，从而显示不同电机的参数值。

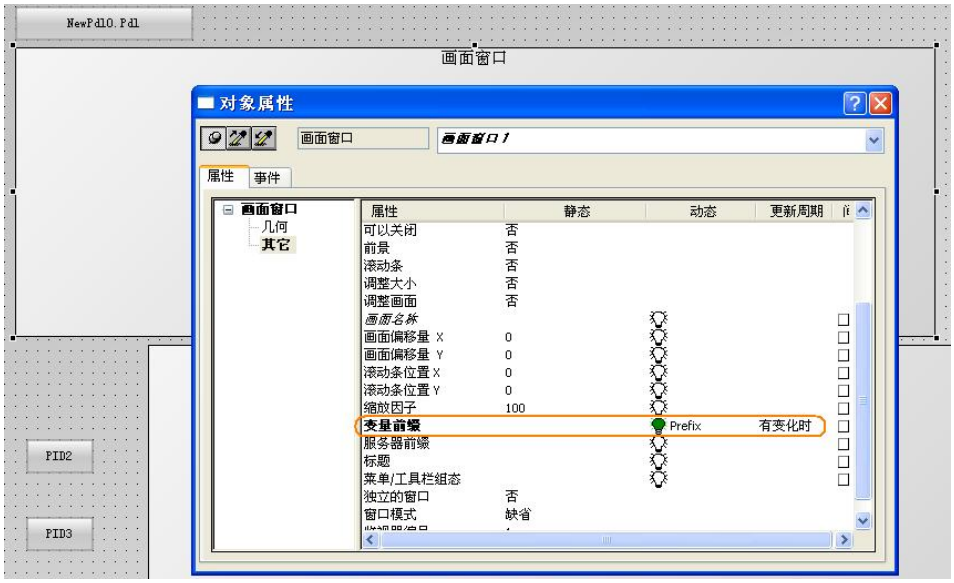

图 18，由变量修改变量前缀

由于改变变量前缀只有在再次装载画面时才起作用。为了使变量前缀的更改起作用，需要给画面窗口添加一个事件，在“变量前缀更改”事件中组态一个直接连接，重新装载画面名称，如图 19。

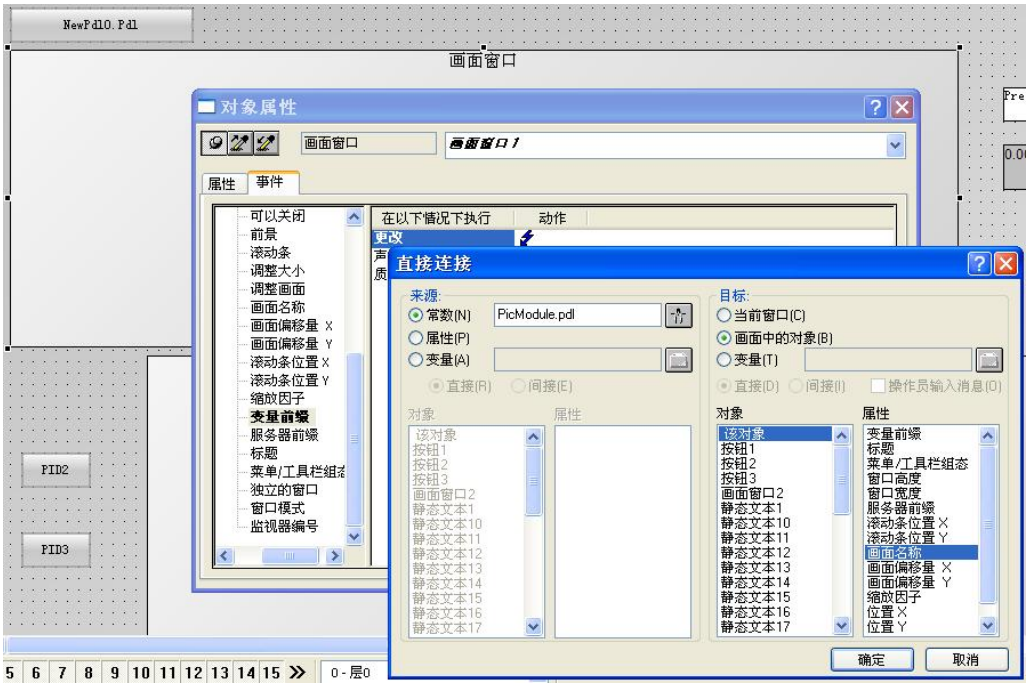

图 19，为画面窗口组态事件

运行后的结果如图 20。

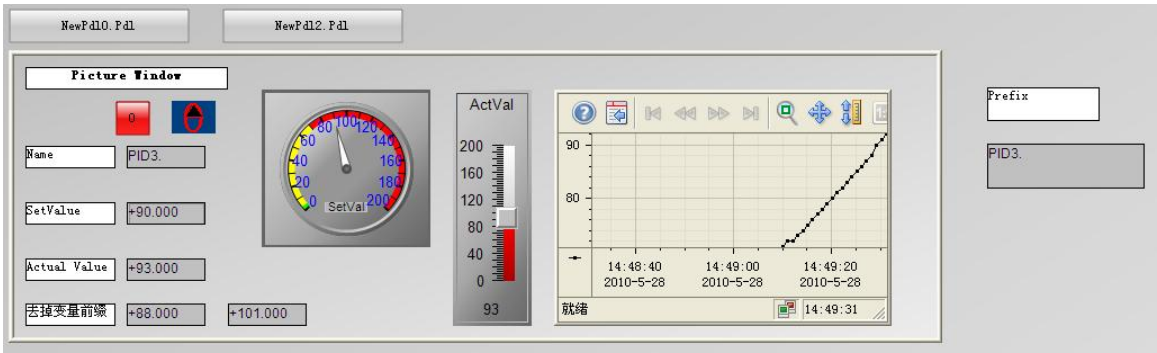

图 20，运行效果

1.2.3 通过脚本修改 TagPrefix

修改变量前缀可以用 C 脚本 SetPropChar(lpszPictureName, “画面窗口 2”, “TagPrefix”, “PID2.”)；然后再重设画面窗口的画面名称，可用 C 脚本 SetPictureName (lpszPictureName, “画面窗口 2”, “PicModule.pdl”)，在按钮”PID2”的鼠标动作中添加 C 动作，如图 21。

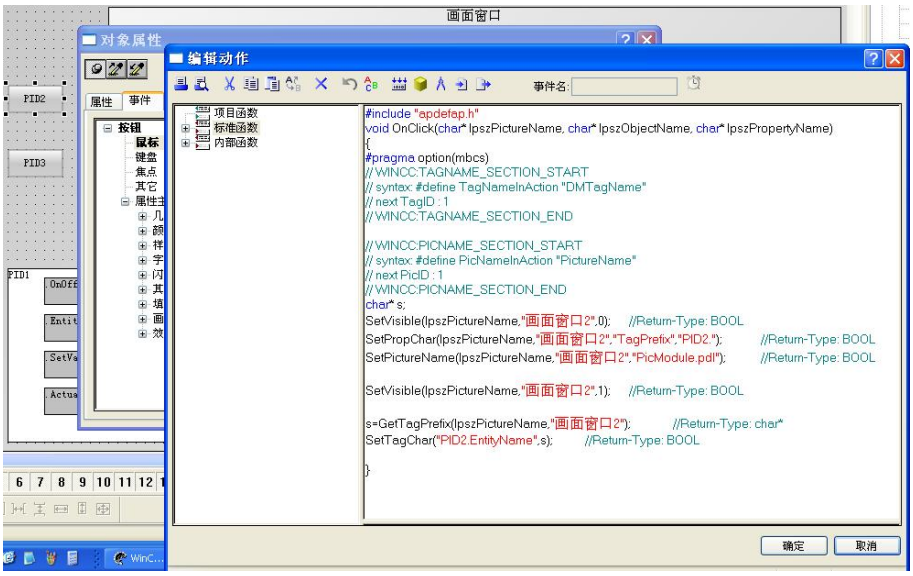

图 21，C 脚本修改变量前缀（1）

运行后，点击按钮 PID2，打开的画面窗口中显示对应 PID2 的各个参数值，如图 22。

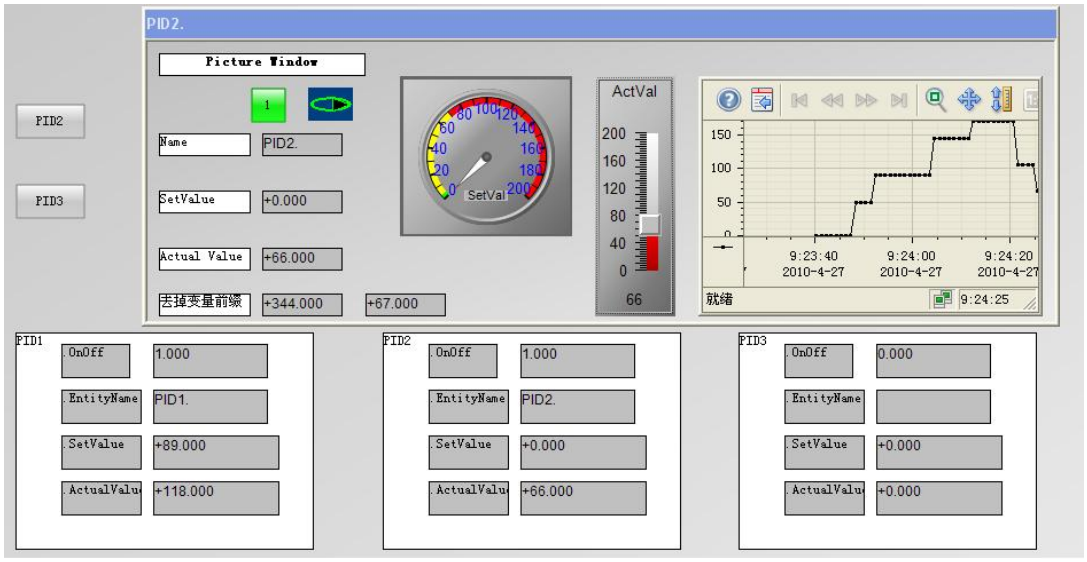

图 22，运行效果（1）

修改变量前缀也可以用 SetTagPrefix (lpszPictureName,”画面窗口 2”, ”PID3.”)；然后再重设画面窗口的画面名称，可用 C 脚本 SetPropChar (lpszPictureName, ”画面窗口 2”, “PictureName”, “PicModule.pdl”)，在按钮”PID3”的鼠标动作事件中添加 C 动作，如图 23。

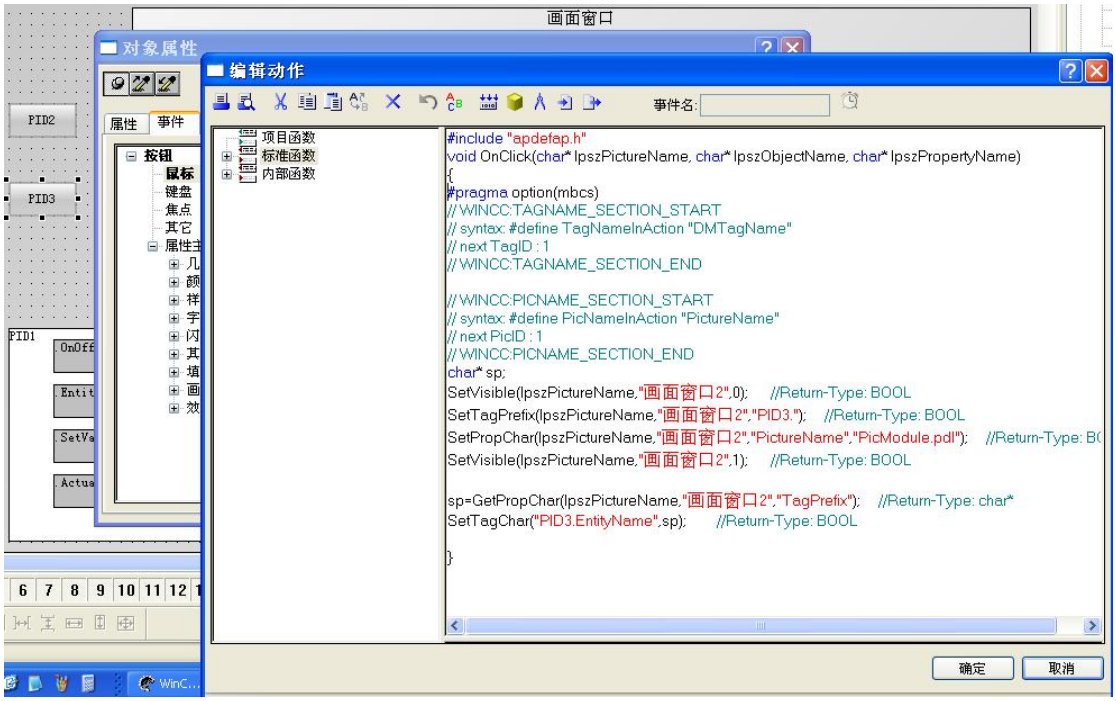

图 23，C 脚本修改变量前缀（2）

运行后，点击按钮 PID3，在打开的画面窗口中显示对应 PID3 的各个参数值，如图 24。

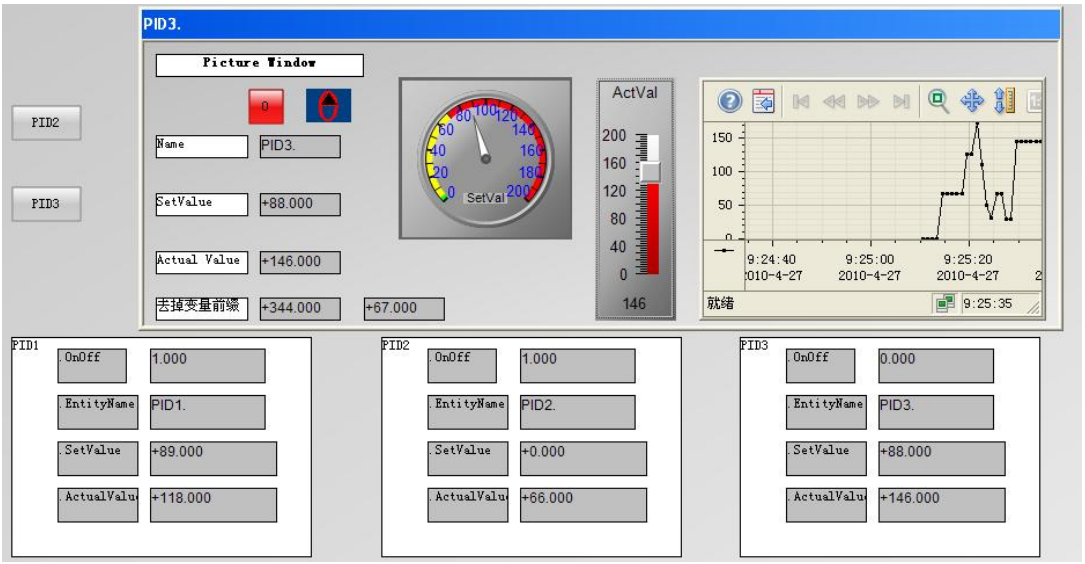

图 24，运行效果（2）

上面描述了如何使用变量前缀和画面窗口的方式实现画面模板，这种方法应用简单，方便掌握，但使用变量前缀的缺陷是它会在画面窗口中所有出现变量的地方都加载变量前缀，如果有些对象的变量不希望添加前缀，该如何处理？

对于 WinCC V7 之前的版本，只能通过脚本、间接寻址、定义全局变量的方法来避免，本文不对此做描述；自 WinCC V7 开始，有更简单的办法取消变量前缀。方法是使用下列变量附加件取消前缀。

“@NOTP”取消变量前缀；

“@NOSP”取消服务器前缀；

“@NOP”取消变量前缀和服务器前缀。

注意：附加件不能用于画面窗口或基本画面的“TagPrefix”或“ServerPrefix”属性。附加件适用于所有动态化类型。

为了说明附加件的使用方法，在画面 PicModule.pdl 中添加一个 IO 域，关联的变量名为 @NOTP::SetValue，如图 25。

图 25，避免变量前缀的组态方法

运行后，这个 IO 域显示的数值为全局变量 SetValue 的数值。

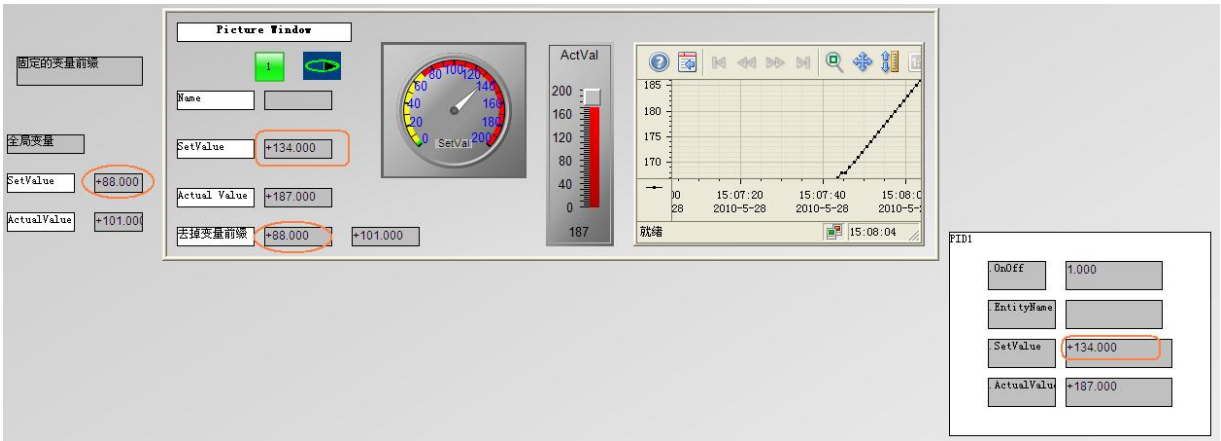

图 26，避免变量前缀的运行效果

2，使用动态向导的画面模板

使用动态向导建立画面模板的方法和上一节所述建立模板画面的方法相同，只是这种方法不需要在画面中连接变量。具体做法如下（同样使用上一节建立的结构类型和变量）。

2.1 创建基准画面

建立一个画面，放置一些对象，如静态文本，输入输出域，按钮等，这些对象都不连接任何变量。保存该画面，如 DynPicModule.pdl。

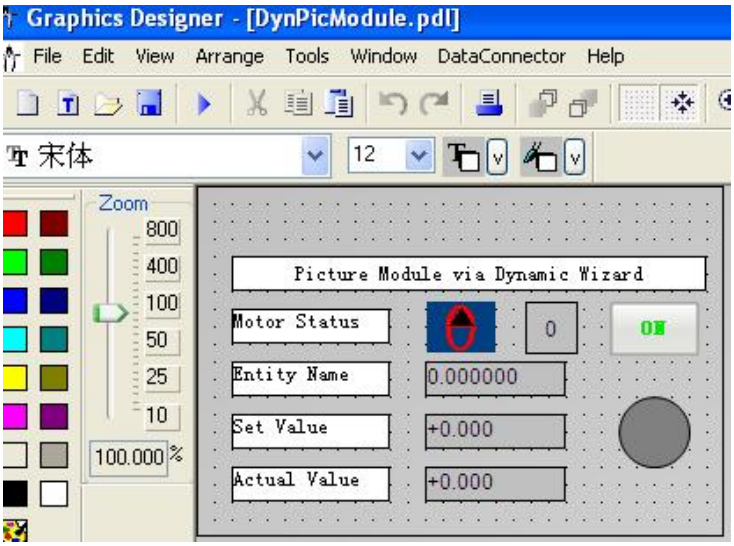

图 27，基准画面的组态

2.2 创建模板

2.2.1 执行动态向导

点击画面中空白处，对这个画面执行动态向导 Picture-Modules 栏下的 Picture-Module template - V1.14，创建一个以当前画面为基础的模板。

2.2.2 选择结构类型

选择将要使用的结构类型，如图 28。

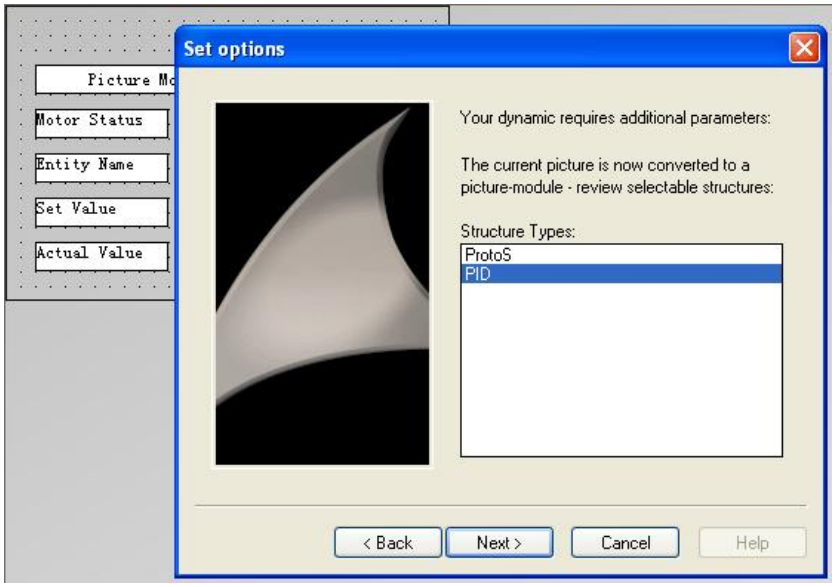

图 28，选择结构类型

2.2.3 连接对象属性

将画面中某些对象的某些属性和结构元素连接上，如将“状态显示”对象的 Index 属性连接到结构元素”OnOff”上，如图 29。

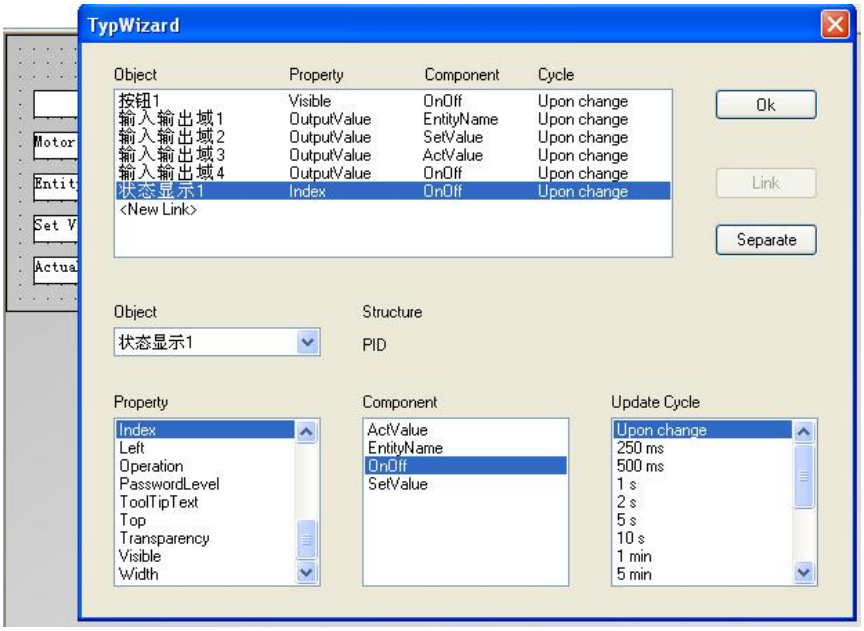

图 29，连接对象的属性

2.2.4 完成向导

WinCC 自动创建一个新画面@Type_DynPicModule.pdl，即模板画面，在这个画面中自动生成一个 IO 域，名为 InstanceName，并组态了 C 动作，如图 30。

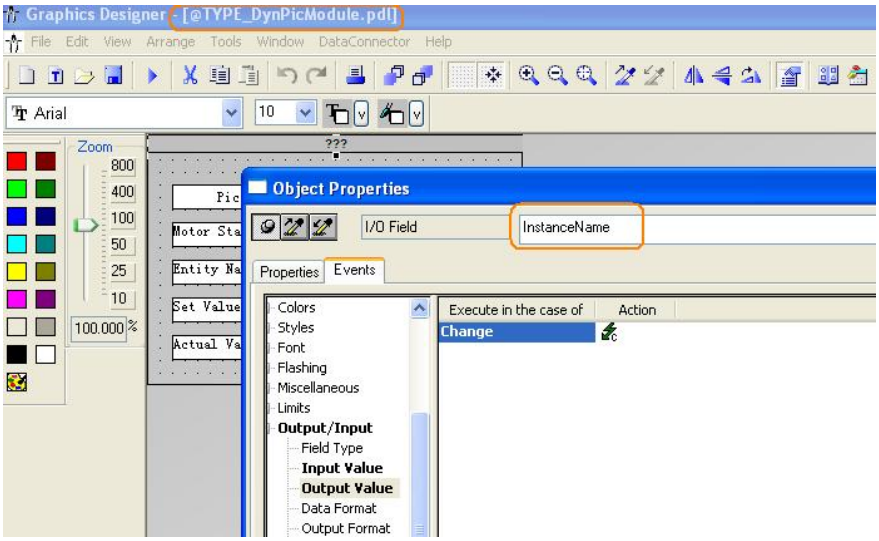

图 30，自动生成的 IO 域---InstanceName

该 IO 域的事件---输出值更改组态的 C 动作如图 31 所示。

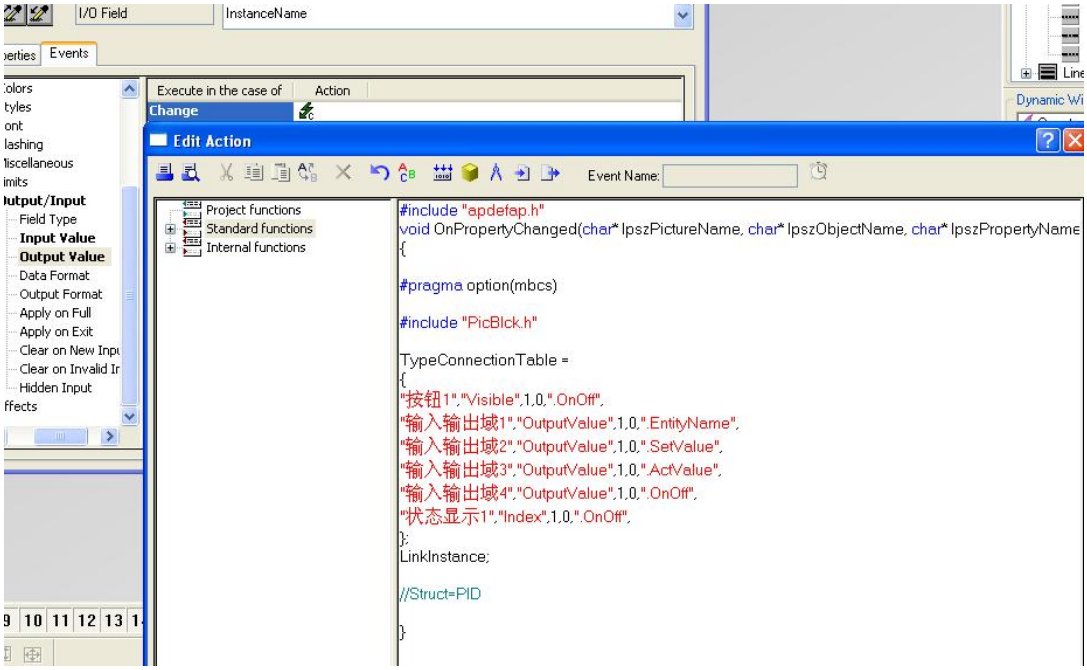

图 31，IO 域关联的 C 动作

脚本的内容是设置画面中对象属性和结构元素的连接关系，这段脚本可以由用户根据自己的需要做改动。但是请**不要**更改 IO 域的名称！

要点：建议在 WinCC 界面语言为英文时调用动态向导 Picture-Module template - V1.14，这样生成的 IO 域名为”InstanceName”，如果在中文界面语言时调用该向导，则生成的 IO 域名为”实例名称”，这时需要另行修改该 IO 域名称为”InstanceName”！

如果已经通过 template 产生了模板@Type_***.pdl，又需要修改，那么如果是修改和结构元素的链接关系，则修改 InstanceName 的“输出值更改”组态的 C 动作；如果要修改画面中的对象或者添加/删除对象，则直接在模板画面中修改即可。

2.3 根据模板生成画面实例

2.3.1 创建一个新画面

再建立一个新画面，在画面中应用动态向导 Picture-Modules 里的 Picture-Module instances-V1.14，在接下来的步骤中选择上一步创建的模板画面，模板的显示方式有四种，这里选择 fixed module in picture，如图 32。

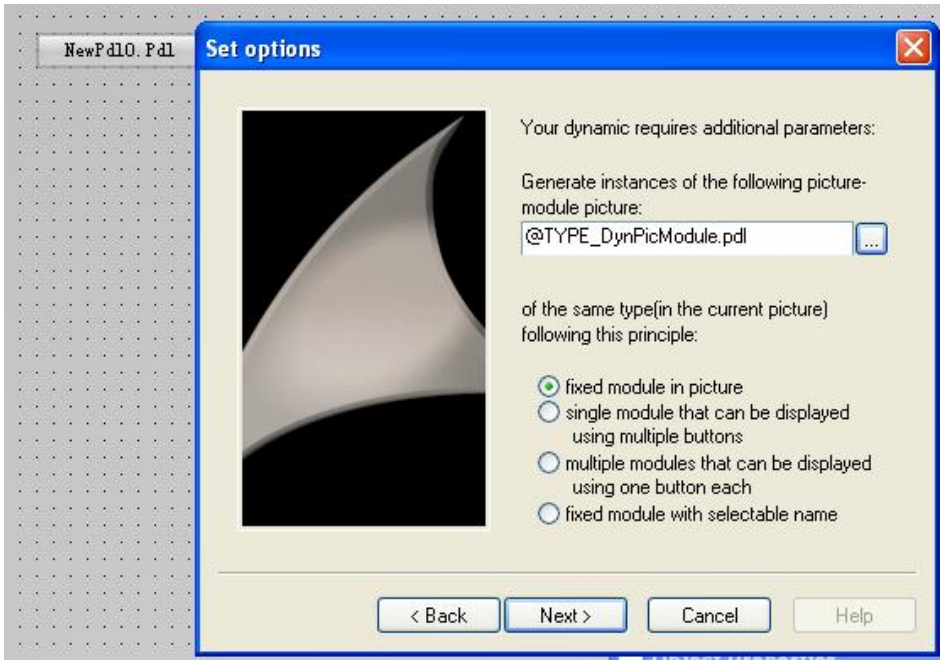

图 32，用模板创建实例

2.3.2 连接模板和结构变量

将模板和结构变量 PID1 关联起来，并设置模板在画面中的位置，如图 33。

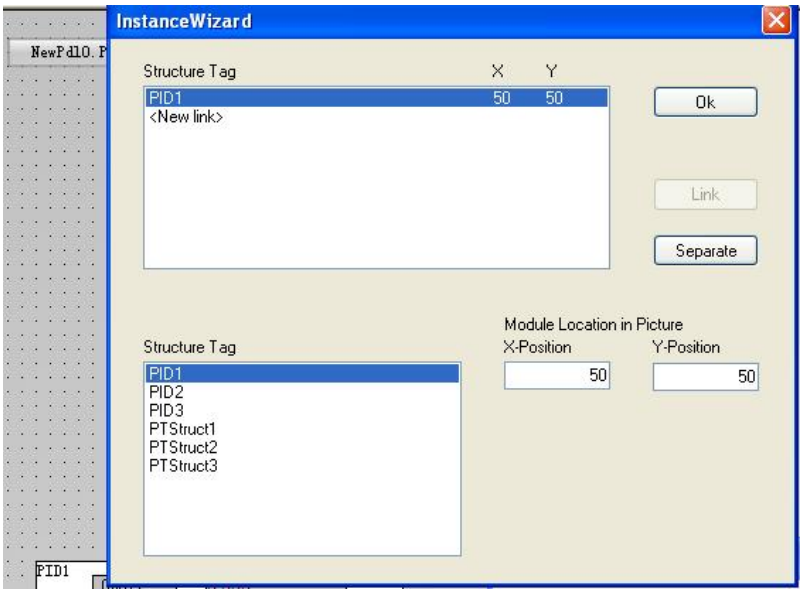

图 33，连接模板和结构变量

2.3.3 完成组态

组态之后，画面中会生成一个画面窗口，由于第（1）步选择 fixed module in picture，画面窗口的画面名称是固定的@Type_DynPicModule.pdl，画面窗口的对象名称为 PID1。

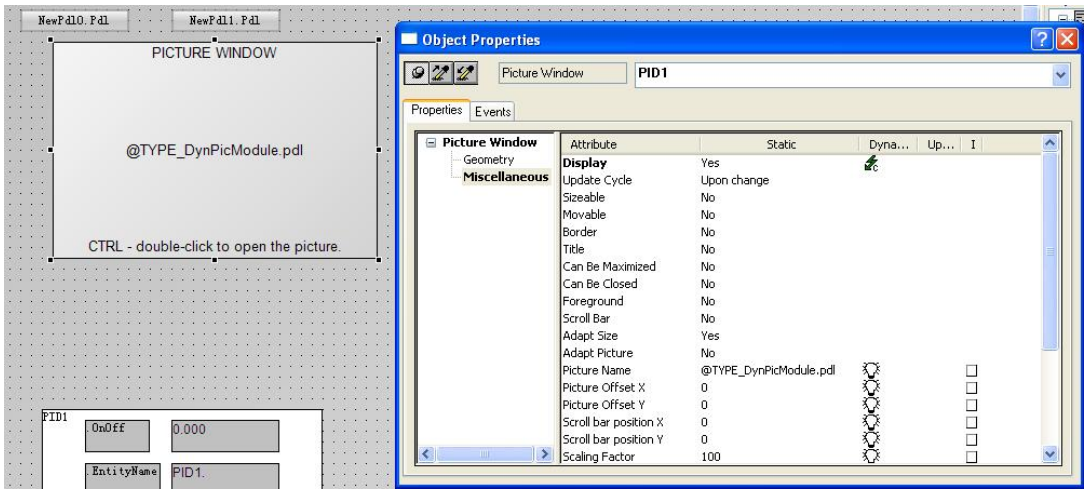

图 34，向导生成的画面窗口

2.3.4 运行效果

组态完成，运行 WinCC 可以看到画面窗口显示的是结构变量 PID1 的各个元素的数值。

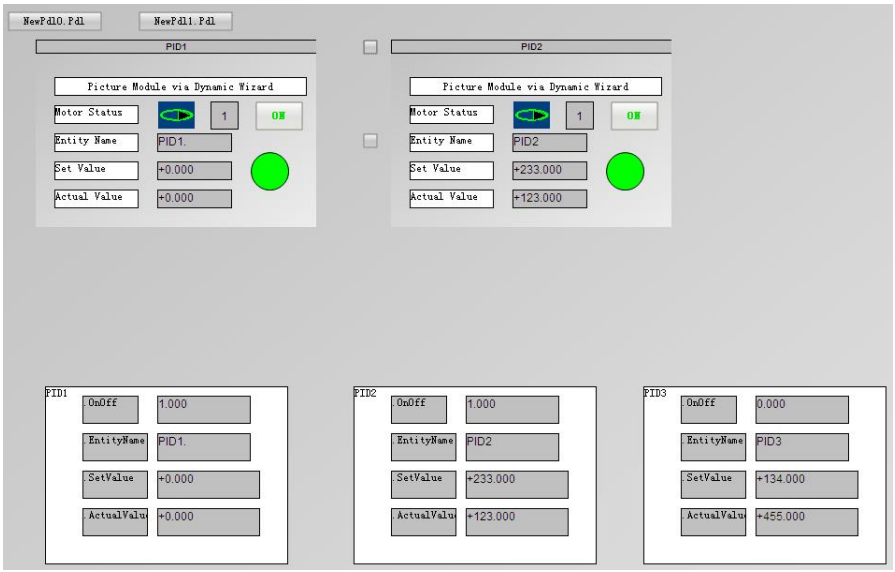

图 35，运行后的效果

2.3.5 其他可能的选择

在步骤 2.3.1 选择模块类型的时候，还有另外三种选择，用户可以自行测试，本文不作赘述。

3，上面两种方法的比较

利用变量前缀，可以将结构元素关联到画面中对象的属性中，也能关联到对象的事件中，包括 C 动作、VBS 动作和直接连接。

利用动态向导生成的模板，只能将结构元素关联到画面中对象的属性里，不能关联对象的事件。但是并不是说模板里面就不能为对象组态事件，只是事件的动作不是直接操作某个变量，而是间接的操作，如利用 C 函数 GetLinkedVariable (lpszPictureName, “IOField1”, “OutputValue”) 定位到”IOField1”所连接的变量。

声 明 本文所描述的是如何利用结构变量组态画面窗口从而降低工作量的方法，但是，并不是说使用画面窗口必须结合结构变量！请明确区别。