

@所有人 汇川 AC702,AC802 电气工程师 5 名，熟悉汇川 PLC，触摸屏，伺服，变频器，机械手等应用，熟练使用 PackML 模板，有面向接口，面向任务设计思想，工价根据能力面谈定价，100 到 150，近期有空，有意向的祥谈..

有人在群里面转发了这样一个招募工程师的信息。

图片中的信息量非常大。我会多次针对其中的话题做出分析。本文针对其中的一点：

面向接口。

说实话，我还真不太了解什么叫面向接口。于是问了一下 AI，给我回答了 2000 字长文，我读后也仍然不太清楚。于是专门问了在 PLC 中的面向接口，回复内容如下：

User: 展示一个 PLC 程序中面向接口的例子

Kimi: 在 PLC（可编程逻辑控制器）编程中，面向接口编程的思想可以通过使用功能块（Function Blocks）和数据类型（Data Types）来实现。虽然 PLC 编程语言（如 IEC 61131-3 标准中的梯形图、功能块图、指令列表等）与面向对象编程语言（如 Java、C#）有所不同，但面向接口的思想仍然可以应用。

以下是一个简单的 PLC 程序示例，展示如何通过功能块和接口来实现面向接口编程。

### 1. \*\*定义接口\*\*

在 PLC 编程中，接口可以通过定义一个通用的功能块来实现。这个功能块定义了所有必要的输入和输出参数，但不包含具体的实现逻辑。

##### 定义通用接口功能块

假设我们需要一个通用的“传感器接口”功能块，所有具体的传感器功能块都将实现这个接口。

```
```plaintext
```

```
// 定义通用传感器接口功能块
```

```

FUNCTION_BLOCK SensorInterface
VAR_INPUT
    Enable : BOOL; // 启用传感器
END_VAR
VAR_OUTPUT
    SensorStatus : BOOL; // 传感器状态
    SensorValue : INT; // 传感器值
END_VAR
```

```

这个功能块定义了所有传感器需要实现的基本功能：`Enable` 输入用于启用传感器，`SensorStatus` 输出表示传感器的状态，`SensorValue` 输出表示传感器的值。

### ### 2. \*\*实现具体的传感器功能块\*\*

接下来，我们定义具体的传感器功能块，这些功能块将实现通用接口功能块的逻辑。

```

##### 温度传感器功能块
```plaintext
// 温度传感器功能块
FUNCTION_BLOCK TemperatureSensor
VAR_INPUT
    Enable : BOOL; // 启用传感器
END_VAR
VAR_OUTPUT
    SensorStatus : BOOL; // 传感器状态
    SensorValue : INT; // 传感器值
END_VAR
VAR
    TempSensor : TempSensorHardware; // 假设的硬件接口
END_VAR
```

```

```

// 实现逻辑
BEGIN
    IF Enable THEN
        TempSensor.Read(); // 读取温度传感器值
        SensorValue := TempSensor.Value;
        SensorStatus := TRUE;
    ELSE
        SensorStatus := FALSE;
    END_IF;
END_FUNCTION_BLOCK
```

```

### ##### 压力传感器功能块

```

```plaintext
// 压力传感器功能块
FUNCTION_BLOCK PressureSensor
VAR_INPUT
    Enable : BOOL; // 启用传感器
END_VAR
VAR_OUTPUT
    SensorStatus : BOOL; // 传感器状态
    SensorValue : INT; // 传感器值
END_VAR
VAR
    PressureSensorHardware : PressureSensorHardware; // 假设的硬件接口
END_VAR

// 实现逻辑
BEGIN
    IF Enable THEN
        PressureSensorHardware.Read(); // 读取压力传感器值
        SensorValue := PressureSensorHardware.Value;
        SensorStatus := TRUE;
    ELSE
        SensorStatus := FALSE;
    END_IF;
END_FUNCTION_BLOCK
```

```

### ### 3. \*\*使用接口\*\*

在主程序中，我们可以通过调用通用接口功能块来使用具体的传感器功能块。这样，主程序不需要知道具体的传感器类型，只需要调用通用接口即可。

```

##### 主程序
```plaintext
// 主程序
PROGRAM MainProgram
VAR
    TempSensor : TemperatureSensor; // 温度传感器实例
    PressureSensor : PressureSensor; // 压力传感器实例
END_VAR

// 启用温度传感器
TempSensor.Enable := TRUE;
// 启用压力传感器
PressureSensor.Enable := TRUE;
```

```

```
// 读取传感器数据
IF TempSensor.SensorStatus THEN
    // 处理温度传感器数据
    // TempSensor.SensorValue
END_IF;

IF PressureSensor.SensorStatus THEN
    // 处理压力传感器数据
    // PressureSensor.SensorValue
END_IF;
```

```

#### ### 4. \*\*面向接口编程的优势\*\*

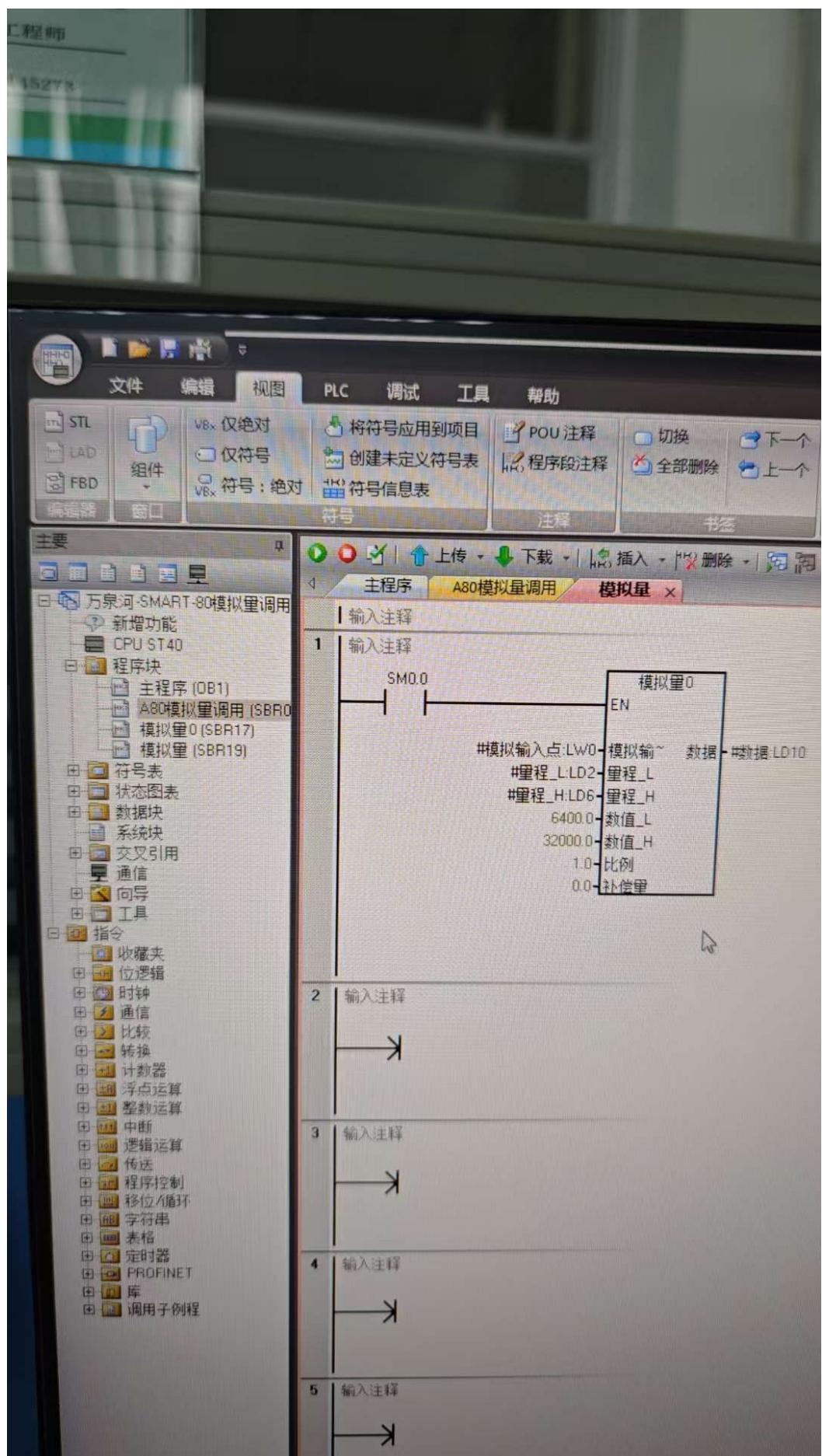
通过这种方式，主程序只需要依赖通用的接口功能块，而不需要知道具体的传感器实现细节。如果需要添加新的传感器类型，只需要实现通用接口功能块的逻辑即可，而不需要修改主程序。

#### ### 5. \*\*总结\*\*

在 PLC 编程中，通过定义通用的功能块接口和具体的实现功能块，可以实现面向接口编程的思想。这不仅提高了代码的可维护性和可扩展性，还降低了耦合度，使得程序更加灵活和易于扩展。

如果你有更具体的需求或场景，我可以进一步调整示例！

不知道读者们有没有看懂。反正我看过之后仍然是一头雾水，不是太有把握真正理解了它的定义。于是，只好用我自己的方式，讲解一下我对面向接口的理解，欢迎读者朋友们批评指正。



昨天另一位烟台方法学员问了我一个问题：

学员：

王总请教您一个问题

学员：

为啥不把所有引脚都做到模拟量 0 这一层呢，直接调用模拟量 0 呢

图片中的程序是我分享的 80 模拟量的程序，所有公众号读者都可以从公众号后台回复关键字：800 获得。

我给他的回复：

这个 SBR:模拟量 0 的块是别人做的，我只是借用了。给改名到 0，又套了一层，得到了一个新的 SBR:模拟量。

相比较之下，新块比原块少了 4 个管脚，你当然可以仍然直接使用原块，但代价是，在有 80 个模拟量数据需要处理的时候，就需要持续调用 80 次，而每次里面也还都要有这 4 个管脚的赋值。而其中的 6400 和 32000 是 AI 卡件 AD 转换的特性决定的。如果调用时需要给他们赋值，除了完全没这个必要以外，还存在一个问题，你把内部的功能需求曝光给了使用方。使用方原本对此参数毫无兴趣，不懂的人一头雾水，万一理解错误填错了数值，还会导致结果错误。总的来说是有百害而无一利。

接口，作为一个交接的界面，英语简写为 UI，其核心思想就是要简洁，内外严格区分。需要外部输入的参数才放到接口。而内部自己能解决的，或者与内部算法相关的参数，都尽量在函数内部自行解决。而不要动辄寻求接口实现。

这是我所认知的面向接口编程。

这种认知很朴素，也没什么高深的原理。所以不管是 IT 程序员还是刚入门级别的 PLC 工程师，都有可能认为太浅，没必要如此郑重其事到还把它当做一种学问。

那么我要说的是，恰恰因为 PLC 行业编程简单，对从业人员的技能要求低，那么对打辅助的辅助人员的要求就要更低，他们客观上就需要你有最极致简洁的设计接口，才可以更容易被辅助人员接受。否则做不到的话，他们就可能以能力不足，搞不懂为由，拒绝帮助你做这些工作，那么所有包括录入等的工作就还是需要自己亲力亲为。

这在 IT 工程师和普通人眼里会认为这样的需求太极致，有些变态。然而那是因为他们不懂的这个行业的特点。

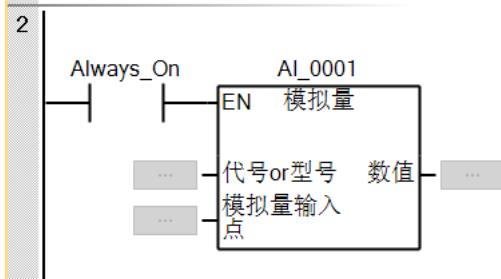
80 模拟量的例子程序中，其实还可以更进一步精简：

现在的程序模块 3 个管脚：#模拟量输入点，#量程\_H，#量程\_L。其中的模拟量输入点对接到模拟量卡件通道，自然不可或缺。然而另外 2 个：#量程\_H，#量程\_L，就值得商榷了。

比如一个项目中 80 个模拟量，分别有压力变送器和温度变送器的测点。然而压力变送器的种类是有限的，比如分别为 0-10Bar 和 0-1000pa，温度测点种类有 0-100 C° 和 0-600C° 各 2 种。我来列一个表：

代号	厂商型号	量程 L	量程 H	量纲
101	AAA-110	0	10	Bar
102	AAA-310	0	1000	pa
201	BBB-100	0	100	C°
202	CCC-666	0	600	C°

然后，我也把精简后的模拟量处理块的接口设计好了：



其中的代号管脚是字符串，规范的设备制造商会为各供应商送来的型号都分配一个专用代号，不管是图纸资料，还是点表，都会有这个代号表示，采购部门也是根据提交的 BOOM 表中的这个代号及数量采购的。

而如果工厂的设计标准化流程没有规划代号，那么也完全可以用厂商的型号名来录入，只不过需要的字符更长一些，也更容易出错，当然也可以有方法处理。本文这里就不提示了。

那么，使用中在程序中完全可以只简单输入代号，就可以实现对一个模拟量信号的标定。由此做到了接口的极简设计。

上述的接口示例是在新版的 SMART V3.0 中演示的，而其实，任何一个 PLC 品牌型号，也都可以实现相同的设计，那么对于应用者来说，就不会存在任何使用难度。而其实，各 PLC 品牌中对模拟量的标定算法是不同的。除了 AD 转换值和精度不同之外，比如三菱的一些旧的 AI 模块，甚至其中还需要复杂的轮询逻辑。但这些都只是封装时一次性的工作。使用中就可以实现完全与模块特性无关了。

这个模块，所有的读者可以自行尝试实现。很简单，但不要因为其简单而懒得动手去做，你只要动动手，就会发现其中还是有一些门道，值得你费一番心思才能把它做到最好。

上面，我举例的温度信号采集都还是假设了使用温度变送器，然而实际工程项目中也还会有一些热电偶、热电阻等信号直接接入专用温度模块，如果有遇到，我们也可以在上述的模块的基础上继续增加功能，实现对这类信号的兼容。

然而，仍然不增加接口。

最后，大家有注意到我给 FB 的接口定义都还是尽量使用了中文，其目的便是在软件平台能支持的情况下，让使用者尽量方便，容易入手。而不至于有某大专毕业实习生来干的时候，因为英语基础差，而抱怨看不懂。

所以，使用中文，让使用者更容易看懂的语言，也是接口优化的一部分。

不给接口添麻烦。

这便是我所认知的面向接口编程。