第 1 章

公 有 元 素

IEC 61131-3 的公有元素包括控制器的基础模型和在 5 种语言中均可使用的公共语言特性，例如，本章依次介绍的数据表示方法、数据类型、变量的声明和引用等。

1.1 控制器模型

控制器基础模型是所有 PAC 都应具备的构成元素、连接关系、通信方式和功能集合，即 IEC 61131-3 的软件模型、通信模型和功能模型。

1.1.1 软件模型

IEC 61131-3 的软件模型描述了 PLC 软件元素及其相互关系，可将一个复杂的软件系统分解为若干个小的可管理部分，并清晰地定义了各部分间的接口方法。其层次结构如图 1-1所示，从最上层到最下层依次是：配置、资源和任务，上层元素可包含下层元素的多个实例。该图也说明了一台 PLC 可以实现多个独立程序的同时装载和运行，并通过任务优先级实现对程序执行的控制。

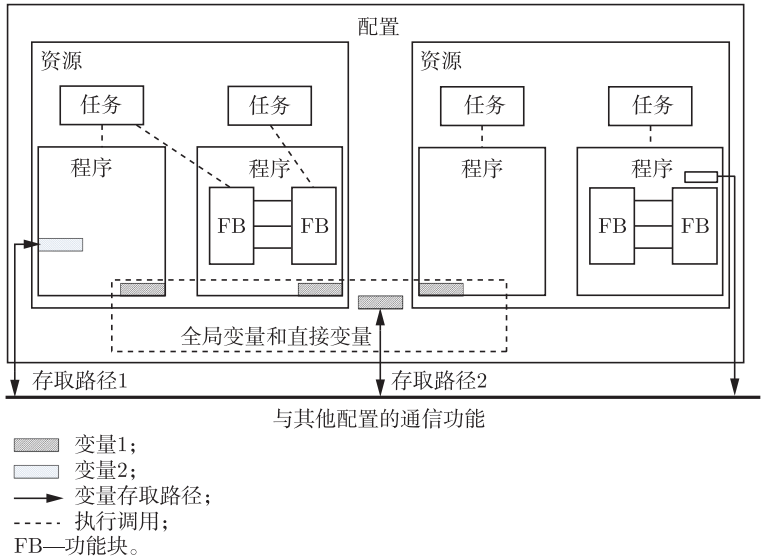


图 1-1 IEC 61131-3 的软件模型 [1]

在一个配置中可以定义一个或多个资源，为程序的执行提供了所需要的资源需求，在一个具有多核处理器的 PLC 中，每个资源可以对应一个处理器核，提供 PLC 的计

算资源，资源中还可以定义 PLC 的全局数据资源。

在一个资源内可以定义一个或多个任务，任务用于规定程序组织单元（Programming Organisation Unit，POU）的运行特性。例如，是周期扫描式执行，还是事件驱动式执行。

任务被配置后可以控制一组程序，程序用 IEC 61131-3 定义的 5 种编程语言来编写，是多种互联的 POU 容器，POU 之间可相互交换数据。函数与功能块是程序内 POU，其内部封装了数据结构和算法。

传统的硬 PLC 模型一般仅包括一个资源，运行一个任务，控制一个程序，且运行于一个封闭系统中，IEC 61131-3 的软件模型则具有以下特点：

（1）适用性强。该软件模型不针对具体的 PLC 系统。

（2）缩放性好。该软件模型既适合小规模系统也适合大型分布式系统。

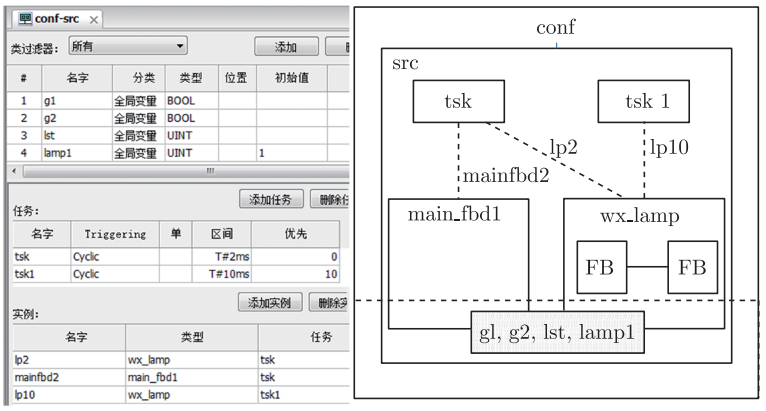
（3）程序并发性高。在一台 PLC 中可同时装载、启动和并发执行多个独立程序。

（4）结构性好。一个复杂程序可以通过分层分解为多层可管理的 POU。

（5）重用性好。函数、功能块、程序可在符合标准的各控制器上重复使用。

（6）任务机制完善。该软件模型保证了 PLC 系统对程序执行的完全控制能力。

IEC 61131-3 编程系统都会具有对标准软件模型进行设置的功能，HPAC 的操作界面如图 1-2（a）所示，可以定义如图 1-2（b）所示的各构成元素，开发者无须（也无法）输入软件模型的结构化文本代码，因为构建时会自动生成这些代码。



（a）　　（b）

图 1-2　HPAC 软件模型操作

（a）操作界面；（b）连接关系

1. 配置

配置的定义从关键字 “CONFIGURATION” 开始，以 “END_CONFIGURATION” 关键字结束，其内部定义了配置级全局变量、资源和任务属性。

2. 资源

资源的定义从关键字“RESOURCE”开始，以“END_RESOURCE”关键字结束，其内部定义了资源级全局变量、任务和程序实例。

3. 任务

任务的定义由关键字“TASK”开始，以“END_TASK”关键字结束，其内部定义了任务名、类型和优先级等信息。HPAC 中任务优先级的范围为 0~99，0 为最高优先级，支持抢占式多任务，高优先级任务可以打断低优先级任务的执行。

HPAC 规定应用的最高优先级必须为扫描任务，并将其扫描周期规定为总线的通信周期。

例 1-1　软件模型编译后的 ST 代码

```
CONFIGURATION conf
  RESOURCE src ON PLC
    TASK tsk(INTERVAL := T#10ms,PRIORITY := 0);
    PROGRAM prg : lamp;
  END_RESOURCE
END_CONFIGURATION
```

例 1-1 中定义了一个名为“conf”的配置，包括名为“src”的资源，运行了一个名为“tsk”的 10 ms 周期任务，优先级为 0，扫描周期 10 ms 也作为总线同步通信周期。“tsk”任务执行名为“prg”的程序实例，其原型来自名为“lamp”的 POU 程序。

4. 全局变量

全局变量被定义在配置或资源内，在配置中声明的全局变量可在这个配置的多个资源中使用，在资源中声明的全局变量可以在资源内多个程序中使用，由于 HPAC 不支持多个资源，所以全局变量定义在资源或配置里没有区别。

5. 实例

HPAC 通过实例建立任务与程序的联系。在图 1-2（a）所示的实例列表中，第一列是实例名；第二列是类型名，在类型栏中可以选择工程中已有的程序；第三列是启动该实例的任务，在任务栏中可选择已有的任务。如果把程序类比为 C++ 语言的类，则实例就是程序的对象，程序只有实例化并与任务绑定才能被执行。

1.1.2　通信模型

IEC 61131-3 定义了图 1-3所示的数据流连接、全局变量连接、通信功能块连接和访问路径连接 4 种通信形式：

（1）数据流连接。图 1-3（a）中直接连接两个功能块的输入和输出接口，就是数据流的数据通信连接，功能块 FB1 中的输出数据变化会改变功能块 FB2 中的输入数据。

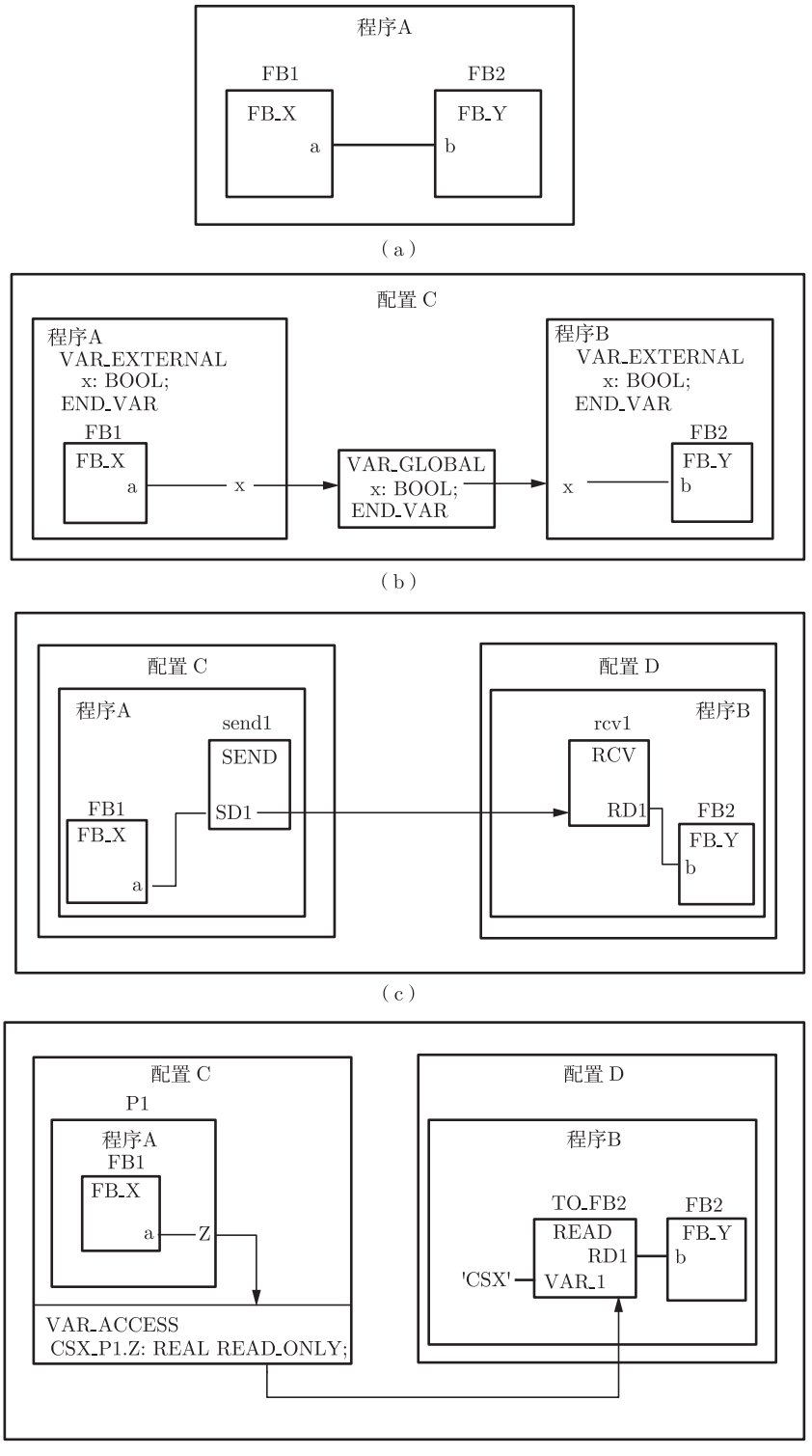


图 1-3　IEC 61131-3 的通信模型 [1]

（a）数据流连接；（b）全局变量连接；（c）通信功能块连接；（d）访问路径连接

（2）全局变量连接。图 1-3（b）中的程序 A 和程序 B 都引用了配置 C 中（也可以在资源中）定义的全局变量 x，程序 A 中对全局变量的修改会影响程序 B 中的值，反之亦然（程序对全局变量的引用见图 1-8）。

（3）通信功能块连接。图 1-3（c）中多个配置间的两个程序可以通过功能块进行通信，程序 A 可将数据发送到另一个配置的程序 B 中。

（4）访问路径连接。图 1-3（d）中多个配置间的程序可通过访问路径实现通信。

HPAC 编程系统只支持与图 1-2（b）类似的单资源、单配置的软件模型，即只支持单 PLC 的单核编程，全局变量定义在配置中或在资源中没有区别。对不同配置间的通信，只能在两个配置上调用通信功能块独立编程后实现，也不支持图 1-3（d）所示的“VAR_ACCESS”关键字和访问路径等。

1.1.3　功能模型

IEC 61131-3 的功能模型描述了可编程控制器系统所具有的功能集合，包括信号处理功能、过程数据接口功能、通信功能、人机界面功能、编程调试功能及电源功能等（见图 1-4），具体功能如下：

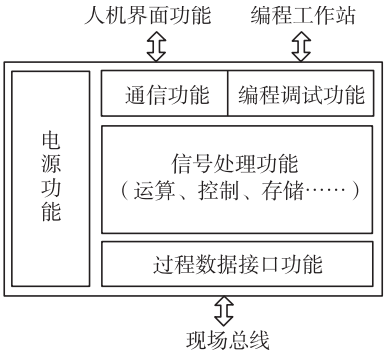


图 1-4　IEC 61131-3 的功能模型

（1）信号处理功能。信号处理功能由应用程序寄存器功能、操作系统功能、数据寄存器功能、应用程序执行功能等组成，它根据应用程序，处理从传感器或内部数据寄存器获得的信号，处理后输出信号送执行器或内部数据寄存器。

表 1-1 给出了最基本的可编程信号处理功能。

（2）过程数据接口功能。HPAC 提供了 BUF_READ 和 BUF_WRITE 等功能块操作总线 I/O 数据区，传感器和执行器通过总线组态工具进行数据区分配，具体内容见 8.2.1 节。

（3）通信功能。HPAC 提供了 HOLD_READ 和 HOLD_WRITE 等功能块操作 Modbus 数据区，实现与其他人机界面（human-machine interface, HMI）及信息化系统的通信集成，具体内容见 8.2.4 节。

（4）人机界面功能。HPAC 提供了 QTOUCH_READ 和 QTOUCH_WRITE 等功能块操作 QTouch 数据区，实现了与内置人机界面组态界面的交互，见 8.2.3 节。

（5）编程调试功能。HPAC 提供了在线调试功能，可观察、强制内部变量的值，并提供了实现单步调试的 PLC_PAUSE 功能块，见 8.2.7 节。

（6）电源等辅助控制功能。在 HPAC 产品体系中，与 H842BS 控制器配套的电源模块具有半分钟的不停电备用电源（Uninterruptible Power Supply，UPS）功能，当电源掉电后应用可进行现场保存处理。

表 1-1　可编程信号处理功能

功能组别		示例
逻辑控制	逻辑	与、或、非、异或、触发
	定时器	接通延迟、断开延迟、定时脉冲
	计数器	脉冲信号加和减
	顺序控制	SFC
数据处理	接口	选择、传送、格式、组织
	模拟量	PID、积分、微分、滤波
	其他系统	通信协议
	人机接口	显示、命令
	大容量存储器	记录
	执行控制	周期执行、事件驱动执行
运算	基本运算	加、减、乘、除、模除
	扩展运算	平方、开方、三角函数
	比较	大于、小于、等于

1.2　标 准 符 号

IEC 61131-3 的符号有语言编码的字符集、标识符、分界符、关键字、空格和注释等。

1.2.1　字符集

IEC 61131-3 规定了 POU 的文本部分，包括文本语言和图形化语言中的文本代码部分，都应采用 ISO/IEC 10646 规范的字符集。目前的 HPAC 中文支持不足，不能使用中文文件夹和带有空格的长文件夹作为项目目录，要尽量避免在代码中使用中文，最好结合代码版本管理工具做好备份和恢复。

1.2.2　标识符

IEC 61131-3 规定，标识符必须由字母、数字、下划线组成，用于区分不同的变量、函数、功能块等语言元素，表 1-2给出了一些标识符的示例。标识符的使用规则如下：

（1）标识符的第一个字符必须是字母或下划线，最后一个字符必须是字母或数字，中间字符只允许是字母、数字或下划线，例如，bT1_ 最后一个字符是下划线，是一个

非法的标识符。

（2）标识符中不区分字母大小写。

（3）下划线是标识符的一部分，但不允许有两个或更多连续的下划线。

（4）标准规定编程系统至少支持 6 个有效位的标识符（在 6 个有效位的编程系统中，AxisID_1 与 AxisID_2 是一样的），HPAC 支持的标识符位数不限。

表 1-2　标识符示例

序号	特性描述	示例
1	大写字母和数字	R123，T43，SZ，DWW
2	大、小写字母，数字，中间的下划线字符	F_W_5，cbd_123，CNN_ABC
3	大、小写字母，数字，开头或中间的下划线字符	_MC，_R_D_431，_CDA_TIE

1.2.3　分界符

分界符是分隔程序语言元素的字符或字符组合的专用字符，表 1-3列出了 HPAC 的分界符及应用示例。

表 1-3　分界符及应用示例

分界符	应用场合	说明和示例
空格	允许在 PLC 程序中插入空格	不允许在关键字、标识符中插入空格
(*	注释开始符号	用户的注释，不允许注释嵌套
*)	注释结束符号	用户的注释，不允许注释嵌套
+	十进制数字的前缀符号	+147
	加操作	1+2
-	十进制数字的前缀符号	-123
	年、月、日的分隔符	D#2018-05-15
	减操作	3-2
#	基底数的分隔符	2#1111_1111
	时间标称的分隔符	T#4ms
.	整数和小数的分隔符	1.0，2.3
	功能块限定符	TON1.Q
e 或 E	实指数分界符	1.0e+6
'	字符串开始和结束符号	'ABCDE'
:	时刻文字分隔符	TOD#15:36:35
:=	赋值操作符	OUTPUT := TRUE;
()	功能块参数表	TON1（IN := %IX1.1，PT := T#50ms);
	子表达式分级	（IN1*（IN3-IN2）+IN4）
	函数参数表	ABS（X1）
,	功能块参数表分隔符	TON1（IN := %IX1.1，PT := T#50ms);
	初始值分隔符	ARRAY（1..2，1..3）OF INT := 1，2，3（4），6;
	被声明变量的分隔符	VAR_INPUT A，B，C: REAL; END_VAR

续表

分界符	应用场合	说明和示例
;	类型分隔符	TYPE W: REAL; END_TYPE
	语句分隔符	OUTPUT1 := A + B; OUTPUT2 := B + C;
%	直接表示变量的前缀	%IX1.1, %QB5
=>	输出连接操作符	TON1 (IN := %IX1.1, PT := T#50ms, Q =>Q1);

1.2.4 关键字

关键字是语言预先定义的标识符，用于定义不同结构、开始或结束特定软件的元素。部分关键字配对使用，如 FUNCTION 与 END_FUNCTION 等，部分关键字单独使用，如 TASK、ABS 等。关键字不能用于任何其他目的，如不能作为变量名或扩展名，例如，不能用 TON 作为变量名，不能用 VAR 作为扩展名等。关键字应遵循标识符的命名规则，不能包含空格。表 1-4 列出了 IEC 61131-3 规定的关键字及其含义。

表 1-4 关键字及其含义

关键字	含义
CONFIGURATION..END_CONFIGURATION	配置段开始、结束关键字
RESOURCE ON..END_RESOURCE	资源段开始、结束
TASK	任务
PROGRAM..END_PROGRAM	程序段开始、结束
PROGRAM WITH	与任务结合的程序
FUNCTION..END_FUNCTION	函数段开始、结束
FUNCTION_BLOCK..END_FUNCTION_BLOCK	功能块段开始、结束
ABS，ADD，GT，BCD_TO_INT 等	标准函数
RS，TOF 等	标准功能块
VAR..END_VAR	内部变量段开始、结束
VAR_INPUT..END_VAR	输入变量段开始、结束
VAR_OUTPUT..END_VAR	输出变量段开始、结束
VAR_IN_OUT..END_VAR	输入、输出变量段开始、结束
VAR_GLOBAL..END_VAR	全局变量段开始、结束
VAR_EXTERNAL..END_VAR	外部变量段开始、结束
ARRAY OF	数组
INT，BOOL 等	数据类型名称
EN，ENO	使能端输入和输出
TRUE，FALSE	逻辑真、逻辑假
TYPE..END_TYPE	数据类型段开始、结束
STRUCT..END_STRUCT	结构段开始、结束

续表

关键字	含义
IF THEN ELSIF..ELSE END_IF	选择语句 IF
CASE OF ELSE..END_CASE	选择语句 CASE
FOR TO BY DO..END_FOR	循环语句 FOR
REPEAT UNTIL..END_REPEAT	循环语句 REPEAT
WHILE DO..END_WHILE	循环语句 WHILE
WITH	与任务结合的 POU
RETURN	跳转返回符
NOT，AND，OR，XOR	逻辑操作符
STEP..END_STEP	步段开始、结束
INITIAL_STEP..END_STEP	初始步段开始、结束
TRANSTION FROM TO..END_TRANSTION	转换段开始、结束
ACTION..END_ACTION	动作段开始、结束

下列功能模块和函数的标识符被保留作为关键字：

（1）标准数据类型名称，如 BOOL、REAL 等。

（2）标准函数名和标准功能块名，即 SIN、COS、RS、SR、TON 等。

（3）指令表语言中的文本操作符，即 LD、ST、DIV 等。

（4）文本语言中的操作符，如 NOT、AND 等。

1.2.5　空格和注释

关于空格和注释的规定有：关键字、标识符、分界符等内不能包含空格；在程序文本里，除了第一种情况，其他任何地方都可以插入空格；程序中允许插入空格的地方都可以添加注释，注释不允许嵌套。表 1-5是空格的示例。

表 1-5　空格示例

特性描述	示例
允许的空格	IF %IX1.2 THEN SB := TRUE；ENG_IF
不允许的空格	I F%IX0.2 TH EN SB := TRUE；ENG_IF

1.3　数 据 标 称

数据外部表示（External Representation of Data）是程序中操作数的表示规则，操作数可以作为常量直接传递给功能块，按类型可分为三类：数值、字符串、时间。在 IEC 61131-3 中被称为三类标称数据“Literal Data”，所谓“Literal”即字面量是指一个字母或符合字面本身所代表的意义，而与它在特定场合的意义无关，例如符号 $，在

特定场合下有特殊的含义（例如在正则表达式里表示某行文本的结尾），但作为字面量“Literal”，它只是代表美元符号，由于这些量的表示方法通常包括了量纲等信息，本书将“Literal”译为“标称”。

1.3.1 数值标称

数值标称（Numeric Literal）按照数据类型分为整数标称和实数标称。按照数制类型分类则可分为二进制、八进制、十进制、十六进制等。十进制实数需要用小数点，布尔型数据可取值为 0 或者 1。

标准规定用数制类型名称和“#”作为前缀表示不同的数制类型（无前缀则为十进制），例如 127 可表示成 2#1111111 或十六进制 16#7F。标准规定数值的数制类型（2,8,10,16）不允许前置分界符（+ 或 −），如 −16#1A 为错误类型，应改为 16#−1A。HPAC 仅支持十进制的负数表示，例如 −127，但对二进制、八进制、十六进制形式表示的负数（例如 2#−010），也会编译出错。

为了提高可读性，数值标称表示时允许使用单一的下划线分段，如 2#111_1111 和 2#1111111 都表示整数 127。表 1-6是数值标称的示例。

表 1-6 数值标称示例

数值标称类型	表示方法	示例
整数	[整数类型名 #] 符号整数或二（八、十、十六）进制整数	INT#−1，UDINT16#A1
	符号整数	−1，1234_5
	二进制整数	2#10_1110 （=46）
	八进制整数	8#123 （=83）
	十六进制整数	16#FF （=255）
实数	[实数类型名 #] 符号实数，实数 [指数]	REAL#2.3，2.1e-2
	符号实数，实数	−2.123，3.654
布尔数	[BOOL#]0 或 1	BOOL#0，0，1
	[BOOL#]FALSE 或 TRUE	BOOL#FALSE，TRUE

在 HPAC 中，所有实数标称的小数点及小数位均不能省略，例如，代码“a:=2.0;”既不能写成“a:=2;”也不能写成“a:=2.;”这两种写法均会产生编译错误。

1.3.2 字符串标称

字符串标称（Character String Literal）分为单字节字符串和双字节字符串两类：单字节字符串的标称以单引号（‘ ’）作为前后标识，如果单字节字符串里出现单引号字符，则需转义表示为 $’，用 $ 与十六进制 ASCII 码可转义为相应的字符，例如，函数 RIGHT（‘a$63’，1）相当于调用 RIGHT（‘ac’，1），返回值为‘c’；双字节字符