设置 PLC 的柔性 IO 地址的探索与实践

王 赛
宜宾职业技术学院，四川 宜宾 644003

[摘要]文章主要围绕 PLC 柔性 IO 地址的设置进行探索实践，通过在 PLC 结构化的程序、UDT 的用户自定义数据类型、间接寻址方式及 PLC 的 SCL 语言的应用等方面开展研究，探索和实践柔性 IO 地址的设置方法和途径。
[关键词]IO 地址；柔性化；PLC；UDT
DOI：10.33142/aem.v2i6.2441 中图分类号：TP273-4 文献标识码：A

The Exploration and Practice of Setting the Flexible IO Address of PLC

WANG Sai
Yibin Vocational & Technical College, Yibin, Sichuan, 644003, China

Abstract: This paper mainly explores and practices the setting of PLC flexible IO address. Through the research on PLC structured program, UDT user-defined data type, indirect addressing mode and PLC SCL language application, the setting method and way of flexible IO address are explored and practiced.
Keywords: IO address; flexibility; PLC; UDT

引言

PLC 控制系统在工业自动化中占主体地位。当前主流的 PLC 都能实现在系统组态时，根据需求，通过组态软件设置好协调的 IO 地址。但是在实际工作中，会经常面临两种情况：一是当系统在实现基本控制的基础上，需要根据用户需求增加控制对象时，需要启用备用的 IO 端口或者扩展 IO 模块。二是因企业的生产流程和工艺的改变，需要对 IO 地址进行重新部署，导致原来已经组态好的 IO 地址发生变化。需要重新调整和分配 IO 地址。以上情况按照传统的方式，都需要通过专业的自动化工程师通过现场调整，重新设置程序参数，重新下载程序，才能够实现。增加了调试的成本，导致生产的效率下降，与当前柔性控制的要求不符。因此本文通过对系统程序开展研究，在不增加其他硬件成本的基础上，实现在系统不停止运行的情况，通过现场的操作人员直接通过人机界面等设备在线修改和设置系统的 IO 参数。灵活应对因系统的简单升级改造，弥补以上的不足，尽可能地降低因停机升级带给企业负面影响，促进生产效率的提高。

1 控制单元程序 FB 块封装方式的研究和实践

1.1 程序结构方式

在 PLC 程序设计中，一般有三种程序结构方式。所有的控制功能都设计到一个程序块内，称为线性化程序；将控制功能分别设计到不同的程序块中，称为分快化的程序；将控制功能固化并封装到一个程序块中，留出对外的接口。使用时直接赋予相关的参数，称为结构化的程序，见图 1。通过对以上三种程序的特点进行分析，线性化和分块化的程序中，所有的 IO 地址都是实际的物理地址，不能实现在线修改，只能够通过修改程序，重新下载的方式。因此要实现对系统的实时修订 IO 地址，必须将程序进行封装，将 IO 地址设置为“形参”，实现结构化编程。

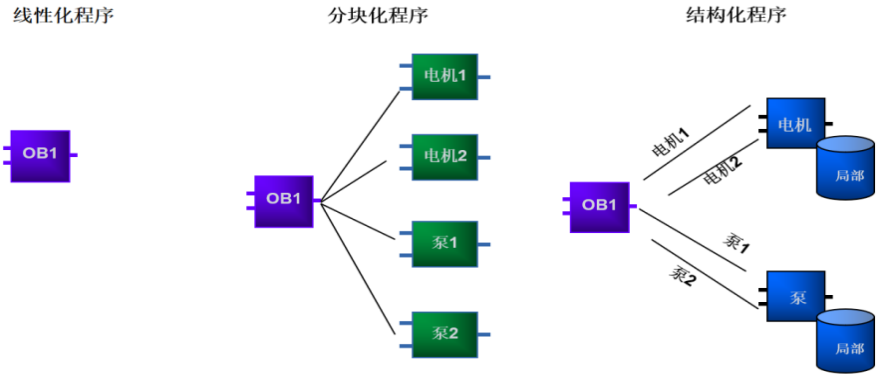


图 1 程序结构的方式

1.2 西门子 S7-1200/S7-1500 程序封装示例

西门子 S7-1200/S7-1500 产品是西门子公司推出的最新 PLC 控制器，本示例就以其组态的博途 V15 版本环境，介绍程序封装的过程。为了便于简化研究，突出重点，以最为简单的电机启动、停止程序为具体的示范，介绍程序封装过程。通过封装后，程序实现了结构化，可以实现反复多次调用该程序，实现相同的功能。

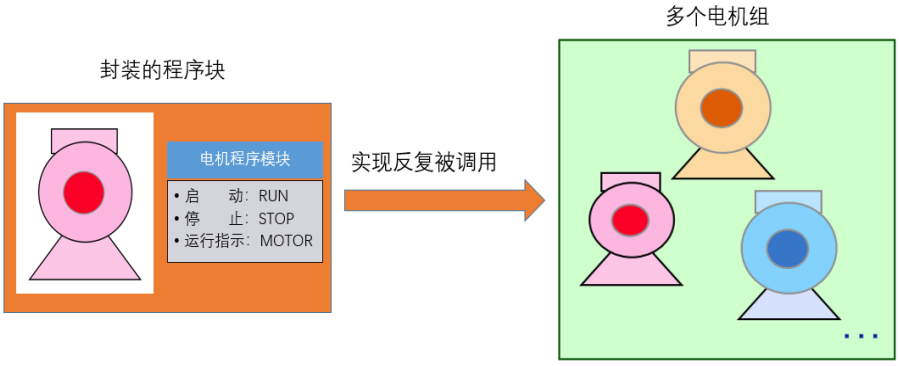


图 2 反复调用封装后的程序

封装过程如下，打开博途 V15 版本软件，新建 PLC 项目，新建程序功能块（FB1 块），进入程序块的组态。在程序块的声明接口参数。并编写程序如图 3 所示。

MOTOR

		Name	Data type
1		Input	
2		RUN	Bool
3		STOP	Bool
4		Output	
5		MOTOR	Bool

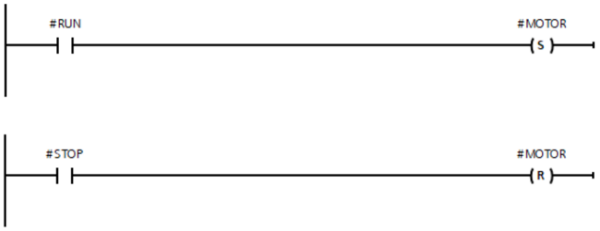


图 3 封装后的 FB 程序块

封装后的 FB1 程序块由主程序实现调用，并在调用时赋予实参，即实际的 IO 地址，如果依然采用此方式，IO 地址实际也是固定的，依然不能实现在线修改，因此必须要寻求其他的途径进行研究。

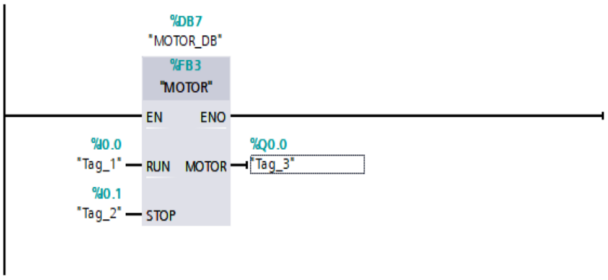


图 4 由 OB 程序块实现传统模式的调用

2 控制单元的 UDT 封装方式的研究和实践

对应固定的控制单元，PLC 还提供有另外一种封装方案，即 UDT（用户自定义数据类型）方式，该方式是将控制对象作为一个单元数据进行定义，定义后的数据单元可以作为数据类型，通用于整个程序块。

2.1 博途模式下 UDT 的创建

UDT 作为一种特殊 PLC 的数据类型，由用户进行定义，定义好的 UDT 数据类型，可以反复被使用。因此是控制单元实现封装的另一种模式。在博途软件中，定义 UDT 数据在项目下的 PLC 数据类型中新建，本例中我们新建“MOTOR_1”的数据块，如图 5 所示。

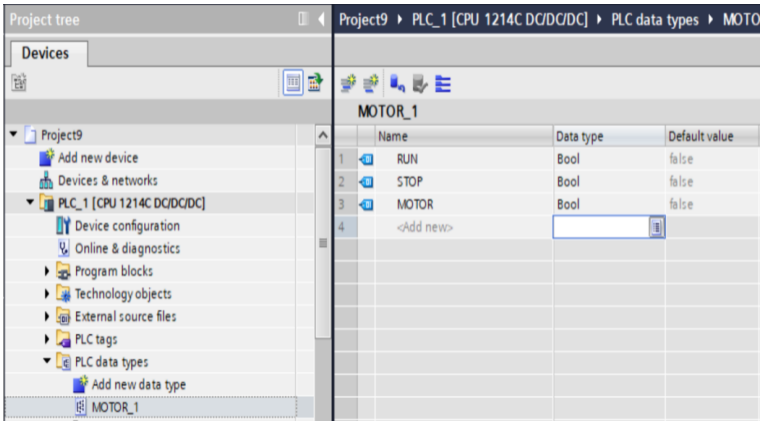


图 5 创建的电机单元的 UDT

2.2 UDT 的应用

UDT 被定义后，可以在程序块中进行使用，在本例中，我们依然通过 FB 块中，通过接口参数中，在静态变量（static）中定义“MOTOR”变量，数据类型选择定义的“MOTOR_1”UTD 数据块，实现对电机的控制，见图 6。但是不难分析使用该方式，依然存在其实没有本质的变化，因为 FB 块的静态变量会由对应的背景 DB 块存储，而背景 DB 块根本不能定义 IO 接口地址，因此不管单纯通过 FB 还是 UDT 这两种方式，是不能实现柔性修订 PLC 的 IO 地址。

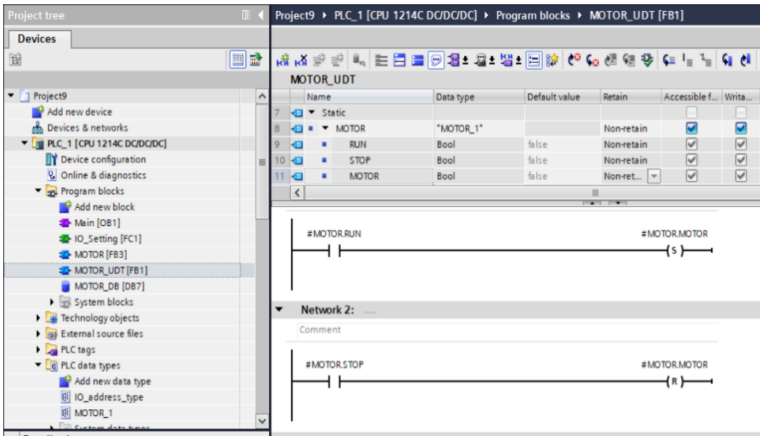


图 6 在 FB 块中使用创建的 UDT

3 对 IO 地址的间接寻址方式的研究和探索

PLC 的间接寻址方式比较多，在西门子上一代产品 S7-300/400PLC 可以通过指针方式实现间接寻址。当前新的 S7-1200/S7-1500 产品的间接寻址的方式较以前发生了比较大的变化。更加易用易学。

3.1 使用 SCL 语言实现间接寻址

编程语言中出现的结构化控制语言（SCL），是一种更高级的编程语言。它以 PASCAL 为基础并可实现结构化编程。除了高级语言元素，SCL 编程语言还将典型的 PLC 元素（例如输入、输出、定时器、标记、块调用等）作为语言元素。通过 SCL 的 PEEK_BOOL 和 POKE_BOOL 指令可以非常便捷地实现对硬件 IO 位地址的间接寻址。指令的主要参数的含义如下：

Area：16#81：表示输入映像寄存器区（I）

16#82：表示输出映像寄存器区（Q）

16#83：表示位存储器区（M）

16#84：表示数据块块区（DB）

DB_number:当 Area:= 16#84，则其值代表数据块的编号，否则为“0”

Byte_Offset：字节地址，设置到字节

Bit_Offset：位地址，具体设置到位
Value：地址的值
Status：16#00：表示 IO 地址未使用
16#01：表示 IO 地址被使用
16#FF：表示错误
下面我们探索通过间接寻址的方式作为突破口，实现 PLC 的 IO 地址柔性设置新方式。

3.2 组态柔性 IO 地址

为了便于调用，将 PEEK_BOOL 和 POKE_BOOL 指令中的变量封装为 UDT 并命名为“IO_address_type”，这样有利于反复使用。

IO_address_type

	Name	Data type	Default value	Accessible f...	Writa...	Visible in ...	Setpoint
1	Area	Byte	16#0	☑	☑	☑	☐
2	Byte_Offset	DInt	0	☑	☑	☑	☐
3	BitOffset	Int	0	☑	☑	☑	☐
4	Value	Bool	false	☑	☑	☑	☐
5	Status	Byte	16#0	☑	☑	☑	☐
6	<Add new>			☐	☐	☐	☐

图 7 “IO_address_type”UDT

在新建的 FC1 程序块中，应用 SCL 语言，编写间接寻址程序，实现对 IO 地址的柔性设置，见图 8。程序首先通过 Area、Value、Status 判断 IO 是否被使用。再通过 Area、Byte_Offset、Bit_Offset 三个条件通过逻辑“与”的结果进行判断，IO 地址的输入输出类型、字节地址、位地址。最后通过对“Area”的赋值，实现定义 I 区还是 Q 区，通过对“Byte_Offset”和“Bit_Offset”的赋值定义具体字节地址和位地址。例如当 Area=16#81，Byte_Offset=10，Bit_Offset=6，则对应实际的物理地址未 I10.6。

```
IF #IO_address.Area = 16#00 THEN
    #IO_address.Value := FALSE;
    #IO_address.Status := 16#00;
    RETURN;
ELSIF (#IO_address.Area = 16#81) AND
    (#IO_address.Byte_Offset >= 0) AND
    (#IO_address.BitOffset >= 0) AND
    (#IO_address.BitOffset <= 7) THEN
    #IO_address.Value := PEEK_BOOL(area := #IO_address.Area,
                    dbNumber := 0,
                    byteOffset := #IO_address.Byte_Offset,
                    bitOffset := #IO_address.BitOffset);
    #IO_address.Status := 16#01;
    RETURN;
ELSIF (#IO_address.Area = 16#82) AND
    (#IO_address.Byte_Offset >= 0) AND
    (#IO_address.BitOffset >= 0) AND
    (#IO_address.BitOffset <= 7) THEN
    POKE_BOOL(area := #IO_address.Area,
        dbNumber := 0,
        byteOffset := #IO_address.Byte_Offset,
        bitOffset := #IO_address.BitOffset,
        value := #IO_address.Value);
    #IO_address.Status := 16#01;
    RETURN;
```

图 8 IO 地址柔性设置程序

4 综合实现 IO 地址柔性设置研究和探索

综合以上的路径，实现对 IO 地址的设置，具体思路如图 10 所示，在设计过程中分别用到了结构化编程、UDT、SCL 间接寻址等方式综合考量才能完成对 IO 地址的柔性设置。

4.1 使用 Array 数组的方式定义 UDT

在控制中可以利用 Array 数组的方式设置 UDT，将之前设置的"IO_address_type"UDT 数据类型作为新建的“MOTOR_Object_1”数组型的 UDT 的数据类型进行设置，对应电机控制单元的启动、停止、及输出 3 个 IO 地址，设计了 3 个数组元素的数组，即 Array[0..2] of "IO_address_type"。每个数组展开后如图 9 所示，对应的数据即可实现 IO 地址的设置。本案例采用数组的方式虽然降低了后期程序设置的可读性，但是提高系统的柔性化，方便推广应用，如图 9 所示。

IO_UDT

	Name	Data type	Default value
1	MOTOR_Object_1	Array[0..2] o...	
2	MOTOR_Object_1[0]	"IO_address_type"	
3	Area	Byte	16#0
4	Byte_Offset	DInt	0
5	BitOffset	Int	0
6	Value	Bool	false
7	Status	Byte	16#0
8	MOTOR_Object_1[1]	"IO_address_type"	
9	MOTOR_Object_1[2]	"IO_address_type"	

图 9 用户定义的设置电机 3 个 IO 地址的数据类型

4.2 使用 FC 程序块为设置 IO 地址

在系统中新建 FC2，打开系统，在参数接口表中设置类型为 InOut 的变量，数据类型选择刚才定义的“IO_UDT”。

MOTOR_Contrl

	Name	Data type	Default value
5	InOut		
6	object	"IO_UDT"	
7	MOTOR_Object_1	Array[0..2] of "IO_a...	
8	MOTOR_Object_1[0]	"IO_address_type"	
9	Area	Byte	
10	Byte_Offset	DInt	
11	BitOffset	Int	
12	Value	Bool	
13	Status	Byte	
14	MOTOR_Object_1[1]	"IO_address_type"	
15	Area	Byte	
16	Byte_Offset	DInt	
17	BitOffset	Int	
18	Value	Bool	
19	Status	Byte	
20	MOTOR_Object_1[2]	"IO_address_type"	
21	Area	Byte	
22	Byte_Offset	DInt	
23	BitOffset	Int	
24	Value	Bool	
25	Status	Byte	

图 10 定义的 FC 程序块的接口参数

通过程序对赋值电机的启动、停止及输出的地址。如图 11 所示。

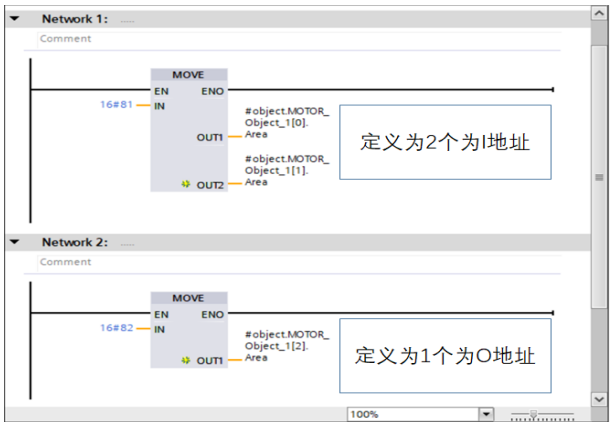


图 11 定义输入输出类型

通过三次调用之前设置的 FC1 程序，分别设置电机的启动的 I 地址、停止的 I 地址及输出的 Q 地址，如图 12 所示。

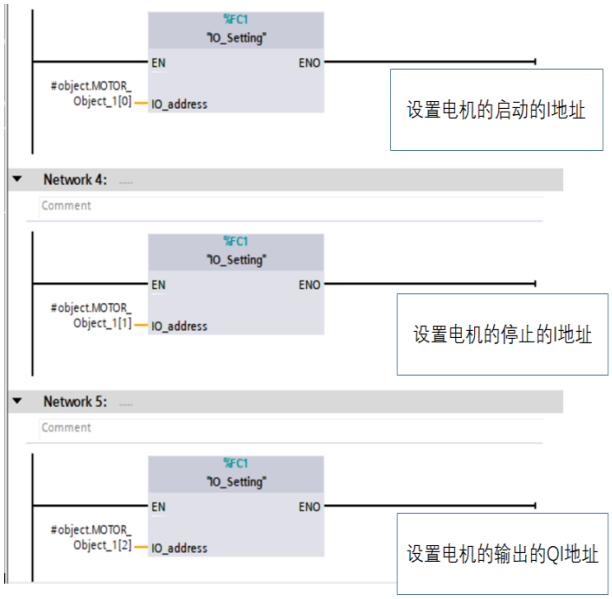


图 12 电机启动、停止及输出地址

通过在 FC2 中调用之前已经设置好的电机控制的 FB1 程序，分别为电机的启动的 I 地址、停止的 I 地址及输出的 Q 地址等“形参”赋“实参”，如图 13 所示。

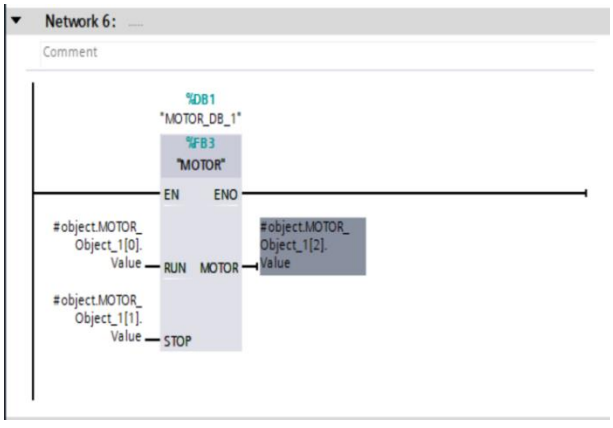
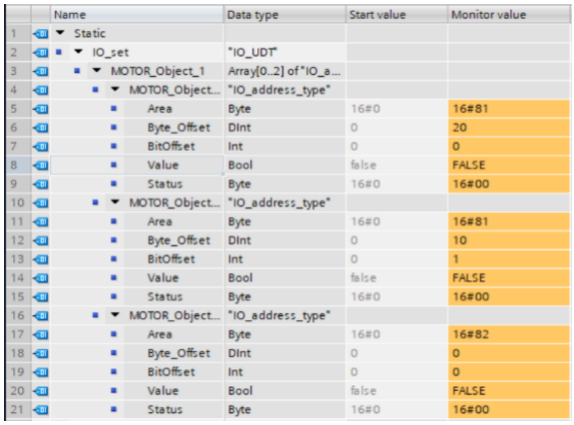


图 13 在 FC2 中调用电机控制的程序块 FB2

4.3 系统的仿真调试

将程序进行仿真，并通过人机界面动态地设置 RUN、STOP 及 MOTOR 的实际 IO 地址，结果与预期一致，说明通过研究和实践，可以实现柔性修订 IO 地址，见图 14。



	Name	Data type	Start value	Monitor value
1	Static			
2	IO_set	"IO_UDT"		
3	MOTOR_Object_1	Array[0..2] of "IO_a...		
4	MOTOR_Object...	"IO_address_type"		
5	Area	Byte	16#0	16#81
6	Byte_Offset	DInt	0	20
7	BitOffset	Int	0	0
8	Value	Bool	false	FALSE
9	Status	Byte	16#0	16#00
10	MOTOR_Object...	"IO_address_type"		
11	Area	Byte	16#0	16#81
12	Byte_Offset	DInt	0	10
13	BitOffset	Int	0	1
14	Value	Bool	false	FALSE
15	Status	Byte	16#0	16#00
16	MOTOR_Object...	"IO_address_type"		
17	Area	Byte	16#0	16#82
18	Byte_Offset	DInt	0	0
19	BitOffset	Int	0	0
20	Value	Bool	false	FALSE
21	Status	Byte	16#0	16#00

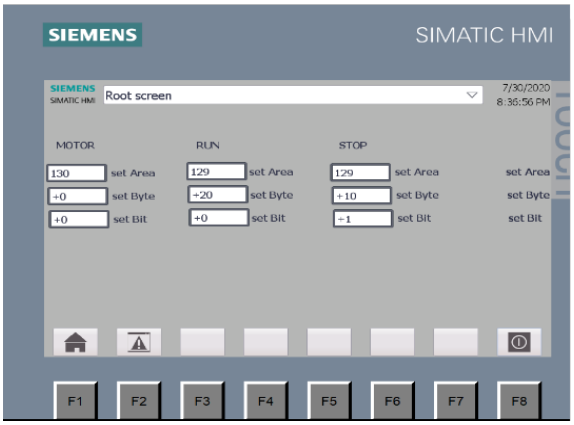


图 14 系统仿真的结果

结束语

本文通过借助西门子的博途平台，使用通过用户程序的设计，在系统不停止工作的情况，直接通过操作人员柔性修改 IO 地址，这种低成本的方式对企业来讲是一种有益的技术创新和实践创新，帮助企业降低改造的成本。同时通过本文的研究，对结构化、柔性化的程序的开发提供了一种新的途径和思路，为技术人员开发相关的程序提供了一种借鉴。因此本研究具有一定的经济价值和技术价值。

[参考文献]

[1]柴瑞娟,陈海霞.西门子 PLC 编程技术及工程应用[M].北京:机械工业出版社,2006.
[2]廖常初.S7-1200/1500PLC 应用技术[M].北京:机械工业出版社,2019.
[3]王一村.PLC 编程应用中的寻址方式[J].电子技术与软件工程,2019,3(23):236-237.
[4]梁斌,吕涛.STEP7 梯形图中实现 DB 块间接寻址的实践[J].北京：柳钢科技,2018,06(7):37-39.

作者简介：王赛（1974.5-），男，毕业院校：四川大学机械工程及其自动化硕士研究生，当前就职于宜宾职业技术学院，职务：项目办主任，宜宾职业技术学院副教授，长期从事自动化行业的研究工作，精通 PLC、变频、工控网等应用技术。